Computational Analysis and Inference of Protein-Protein Interactions from Domain Information

ドメイン情報からの タンパク質間相互作用の解析と予測

Morihiro Hayashida

林田　守広

Computational Analysis and Inference of Protein-Protein Interactions from Domain Information

ドメイン情報からの タンパク質間相互作用の解析と予測

Morihiro Hayashida

林田　守広

Abstract

As sequences for the whole genome of a considerable number of organisms have become available, many researchers have focused on understanding functions of genes and proteins. Information about protein-protein interaction is indispensable for understanding protein functions since protein-protein interaction plays a fundamental role in many cellular processes such as regulation of transcription and translation, signal transduction, and recognition of foreign molecules.

Several computational methods have been proposed for inferring protein-protein interactions. Deng et al. proposed a probabilistic model of protein-protein interactions based on domain-domain interactions, and developed an inferring method using an EM algorithm from this model. They found some biologically significant novel interactions. However, the classification accuracy of their method is not so high. Therefore, I propose a new method based on linear programming, and improve the accuracy. Advantages of linear programs are that we can solve them efficiently and can add several kinds of constraints. I show that the proposed method outperforms existing methods.

On deriving algorithms for the inference problem, it is essential to understand how difficult the problem is. Even though various methods for the problem have been already proposed, it has not been analyzed rigorously from a computational point of view. I hence define a problem to maximize correctly classified examples, and prove the problem is MAX SNP-hard, which also means the problem is NP-hard. Moreover, it means that there is no polynomial-time algorithm to approximate the problem by an arbitrary ratio. Therefore, heuristic algorithms such as the proposed method are required.

Recently, large-scale biological two-hybrid systems were developed for

comprehensive analysis of protein-protein interactions. Though these experiments revealed many unknown interactions, there was a large gap between the results of several groups for same species. In a group, experiments were performed for each protein pair multiple times, the number of observed interactions for each protein pair was counted, and whether each protein pair interacts or not was determined using a threshold. However, among protein pairs which were observed to be less than the threshold, there can be actually interacting protein pairs. Therefore, it is reasonable to use the ratio (strength) of the number of observed interactions to the number of experiments, and to infer that for unknown protein pairs, rather than to deal with whether a protein pair interacts or not.

Thus, I propose a new method for inferring strengths of protein-protein interactions from such experimental data. This method tries to minimize the errors between the ratios of observed interactions and the predicted probabilities in training data, where this problem is formalized as a linear program based on the probabilistic model.

In addition, I propose a simple method for inferring strengths of protein-protein interactions to improve the running time of the LP-based method. I show that the proposed methods outperform existing methods, and in addition, the simple method is much faster than the LP-based method. Moreover, I apply the LP-based method to biological experimental data, and show that the biological result is improved.

Recently, many researchers have studied biological networks, and showed that their networks are scale-free. This is also true for protein-protein interactions, an interaction network in which a vertex that is a protein is known as scale-free.

In order to understand how proteins have obtained various functions, I analyze a network of proteins using domain information. I consider differences of domain compositions between proteins, and introduce a protein domain network. This network also shows a scale-free behavior. I propose a model of protein evolution using domains, and show that the model can reconstruct the protein domain network.

Acknowledgments

I would like to express my utmost gratitude to my thesis supervisor, Prof. Tatsuya Akutsu, for his guidance and advice. I became interested in computer science and bioinformatics under his tutelage. Without his encouragement, I would not have finished this thesis. I would like to express my sincere gratitude to the other members of my committee, Prof. Osamu Gotoh and Prof. Shigeo Kobayashi, for their valuable comments and suggestions.

I would also acknowledge Dr. Nobuhisa Ueda for his helpful comments and warm support. I appreciate his willingness to discuss my methods and results on inferring protein-protein interactions. I would also like to thank Dr. Jose Carlos Nacher Diez for his helpful suggestions and encouragement. I also appreciate his willingness to discuss my models for protein domain networks.

Thanks also go to my current and past colleagues, especially, Dr. Setsuro Matsuda and Masaki Yamamura for their encouragement and entertainment, and John Brown for his correction of this thesis.

Finally, I wish to express my deep gratitude to my parents and family for their support through my entire life.

Publication Notes

Chapter 2 is based on the paper [18] in the *Bioinformatics Journal.* It was also presented at the *European Conference on Computational Biology 2003.*

Chapter 3 is based on the paper [20] in the *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences (Japanese Edition).* It is an extended version of [19] which was presented at the *International Workshop on Bioinformatics and Systems Biology 2004.*

Chapter 4 is based on the above two papers [18, 20].

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Due to rapid progress of the genome sequencing projects, whole genomic sequences of more than several tens of organisms have already been determined. As a next step of the genome projects, many researchers focus on understanding of functions of genes and/or proteins. Information about protein-protein interaction is important for understanding of protein functions because protein-protein interaction plays a key role in many cellular processes.

Several computational methods have been proposed for inference of protein-protein interactions. Enright et al. [16] and Marcotte et al. [32] proposed the gene fusion/Rosetta stone method. Their method finds pairs of proteins each of which putatively interact if each of them is encoded separately as a distinct gene in an organism, and they are fused in another organism. Marcotte et al. [33] also proposed a method combining multiple sources of data such as proteins evolved in a correlated fashion and correlated messenger RNA expression patterns. Wojcik et al. [42] proposed the interaction domain pair profile method. Gomez et al. [17] proposed probabilistic models to form a network of protein-protein interactions based on probabilities of interactions (attractions and repulsions) between domains. Mamitsuka [31] proposed a probabilistic model called the hierarchical class model to predict protein-protein interactions from information on protein classes. Bock et al. [10] applied the SVM (support vector machine) [12] to inference of protein-protein interactions. Lu et al. [30] developed a prediction method

called MULTIPROSPECTOR based on a threading algorithm, and which is able to identify the residues that participate directly in an interaction between proteins.

Recently, some methods were proposed for inferring domain-domain interactions (and/or signature-signature interactions) from protein-protein interaction data. Domain-domain interaction data are useful not only for more detailed understanding of protein-protein interactions but also for inferring protein-protein interactions: two proteins are expected to interact if these proteins contain an interacting domain pair(s). Sprinzak and Margalit proposed the association method for computing the score for each domain pair [37]. Kim et al. [27] proposed similar scores and applied the scores to inference of protein-protein interactions. Deng et al. [14] proposed an EM (Expectation-Maximization) algorithm for estimating the probability of interaction for each domain pair. They compared the EM method with the association method using protein-protein interaction data by Uetz et al. [38] and Ito et al. [23, 24], and showed that the EM method was better than the association method. Moreover, they found some biologically significant novel interactions such as interactions between CTT1 and PEX14, between TAF40 and SPT3, and between RPSOA and APG17. However, the classification accuracy is not so high.

Therefore, in chapter 2, I propose a new method based on linear programming for inferring protein-protein interactions under the framework of domain-domain interactions. We call this method LPBN (Linear Programming-based method for BiNary interaction data). In order to minimize the errors of classification, I use a technique similar to robust linear programming [8] and soft margin [12].

An advantage of using linear programs is that we can solve them efficiently. In addition, we can add several kinds of constraints such as ranges of probabilistic variables, and thus easily can combine the method with other methods. The LPBN method is compared with the association method [37], the EM method [14] and the SVM-based method using real protein-protein interaction data. It is shown that the LPBN method outperforms other methods.

On deriving algorithms for the inference problem, it is essential to un-

derstand how difficult the problem is. Even though various methods for the problem have been already proposed, it has not been analyzed rigorously from a computational point of view.

In chapter 3, I hence define a problem to maximize correctly classified examples, and prove that the problem is MAX SNP-hard, which also means the problem is NP-hard. Moreover, it means that there is no polynomial-time algorithm to approximate the problem by an arbitrary ratio. Therefore, heuristic algorithms such as the proposed LPBN method are required.

Recently, large-scale biological two-hybrid systems were developed for comprehensive analysis of protein-protein interactions in *Saccharomyces cerevisiae* (budding yeast) [23, 24, 38]. Though these experiments revealed many unknown interactions, there was a large gap between the results by Ito et al. [23, 24] and Uetz et al. [38].

Ito et al. [23, 24] performed multiple experiments for each of the protein pairs. However, the results were not always the same for the same pair. They counted the number of observed interactions for each protein pair, and called it IST (Interaction Sequence Tag) hits. Protein pairs which have equal to or more than three IST hits were considered as interacting pairs, and were published.

However, there were pairs that actually interacted even though they had less than three IST hits. Therefore, it is reasonable to use the ratio of the number of observed interactions (IST hits) to the number of experiments as input data, where the ratio is also referred to as the *strength* in this thesis.

In chapter 4, I propose a new method for inferring strengths of protein-protein interactions based on domain-domain interactions. This method tries to minimize the errors between the ratios of observed interactions and the predicted probabilities in training data. I formulate this minimization problem by using linear programming in a similar way to the LPBN method. We call this method LPNM (Linear Programming-based method for NuMerical interaction data). It is shown that the method is comparable to existing methods for binary data such as the EM method, and outperforms them with respect to the errors of strengths between real data and predicted strengths.

However, since the LPNM method is based on the linear programming approach, it may require a large amount of time to infer strengths for a large

data set.

In chapter 4, I also propose a simple method to infer the strengths of protein-protein interactions based on the association method by Sprinzak et al [37]. In an experiment with a data set of protein-protein interactions in yeast, it runs more than 150 times faster than the LPNM method, and achieves almost the same accuracy.

In previous chapters, based on interactions between domains, I have proposed several new methods for interaction between proteins and their interaction strengths. Recently, many researchers have studied characteristics and features of some biological networks from a graph theoretical point of view, and found out that some networks are scale-free. For example, the network of protein-protein interactions of Saccharomyces cerevisiae by Jeong et al.[25] and metabolic networks by Wagner et al.[41] are scale-free. Scale-free networks have scaling properties that the probability that a vertex in the network has a degree k decays as a power law with a negative exponent.

In order to understand how proteins have obtained various functions, in chapter 5, I analyze a network of proteins using domain information. I consider differences of domain compositions between proteins, and define a protein domain network. This network also shows a scale-free behavior. In addition to a negative power-law behavior, it shows a positive power-law behavior. I propose a model to reconstruct the network, and show theoretically that the model generates two types of power laws. Moreover, I verify them through some computational experiments.

Finally, in chapter 6, I give conclusions of this thesis and mention some possible future work.

Chapter 2

Inferring Protein-Protein Interactions from Domain Information

In this chapter, I propose a new method and some combination methods for inferring protein-protein interactions using domain information. Before I describe these methods, we review domain information and the probabilistic model of protein-protein interactions proposed by Deng et al. [14] which we use to derive my methods, and also review methods to infer interactions between proteins: the association method by Sprinzak et al. [37] and the EM method by Deng et al. [14]. The EM method also uses the probabilistic model. We will compare my methods with these methods to confirm the abilities of my methods.

2.1 Domain Information

Domains are functional or structural modules in proteins, and are organized by cohesion between sidechains which stabilize unique structures. They are further stabilized by folding around metal centers, by forming some disulfide bonds, or are a case of short repetitive units. Most proteins contain two or more domains although many proteins are single domains. For instance, Figure 5.25 in chapter 5 shows the distribution of the number of domains of

Homo sapiens.

Domains are identified using characteristic sequence motifs, profiles or fingerprints. These sequence data are stored in some databases such as InterPro [45], Pfam [7] and SMART [29].

2.2 Definition of Probabilistic Model of Protein-Protein Interactions

Let $P_1, \ldots, P_N$ be proteins. We also use P_i to denote a set of domains in P_i. Let $D_1, \ldots, D_M$ be domains in proteins $P_1, \ldots, P_N$. For notational convenience, P_{ij} and D_{mn} represent the protein pair (P_i, P_j) and the domain pair (D_m, D_n), respectively. Let $\mathcal{P}$ be a multi set of protein pairs P_{ij}. We also use P_{ij} to denote a set of domain pairs between P_i and P_j (i.e., $P_{ij} = \{D_{mn} | D_m \in P_i, D_n \in P_j\}$).

In this probabilistic model, an interaction between P_i and P_j (one between D_m and D_n) is represented as a random variable P_{ij} (D_{mn}). P_{ij} takes 1 if P_i and P_j interact with each other, otherwise $P_{ij} = 0$. In the same manner, $D_{mn} = 1$ if D_m and D_n interact with each other, otherwise $D_{mn} = 0$. This probabilistic model assumes that domain-domain interactions are independent and two proteins interact if and only if at least one domain interacts with a domain from another protein (see Figure 2.1). Under this assumption, the probability that P_i and P_j interact with each other is given by

$$\Pr(P_{ij} = 1) = 1 - \prod_{D_{mn} \in P_{ij}} (1 - \lambda_{mn}), \tag{2.1}$$

where λ_{mn} denotes the probability that D_m and D_n interact with each other (i.e., $\lambda_{mn} = \Pr(D_{mn} = 1)$).

2.3 Algorithms

In this section, I describe the association method [37], the EM method [14], and the proposed LP-based method along with its variants. I also propose a simple SVM-based method.

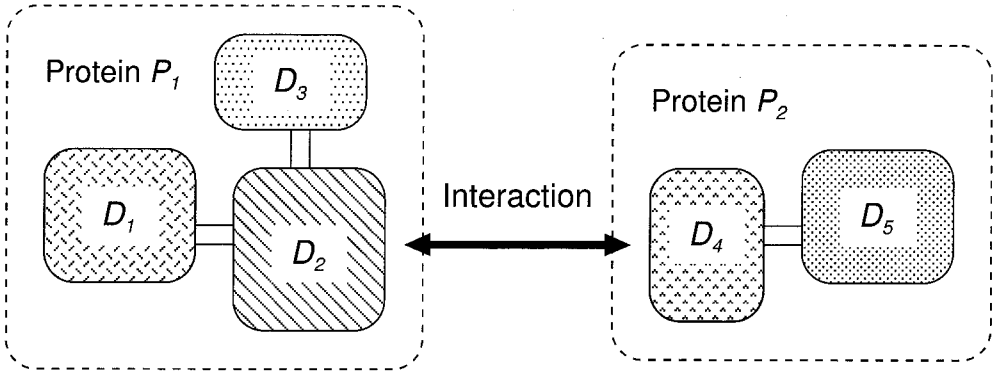

Figure 2.1: Inference of protein-protein interactions through domain-domain interactions. In this case, we infer that proteins P_1 and P_2 interact with each other since domains D_2 and D_4 interact with each other.

2.3.1 Association Method (Sprinzak et al., 2001 [37])

The association method assigns a simple score to each domain pair (D_m, D_n). Let N_{mn} be the number of protein pairs (in the training data set) containing domain pairs (D_m, D_n). Let I_{mn} be the number of interacting protein pairs (in the training data set) containing domain pairs (D_m, D_n). The score (probability of interactions) for (D_m, D_n) is simply defined by

$$ASSOC(D_m, D_n) = \frac{I_{mn}}{N_{mn}}. \tag{2.2}$$

2.3.2 EM Method (Deng et al., 2002 [14])

The EM method uses the probabilistic model in Section 2.2. In this probabilistic model, the probability that P_i and P_j interact with each other is given by

$$\Pr(P_{ij} = 1) = 1 - \prod_{D_{mn} \in P_{ij}} (1 - \lambda_{mn}). \tag{2.3}$$

Deng et al. [14] considered two types of experimental errors: false positives, in which two proteins do not interact in reality but were observed to be interacting in the experiments, and false negatives, in which two proteins interact in reality but were not observed to be interacting in the experiments.

Let fp and fn denote the false positive rate and the false negative rate, respectively. Letting O_{ij} be the variable for the observed interaction result for P_i and P_j ($O_{ij} = 1$ if the interaction is observed), we have:

$$fp = \Pr(O_{ij} = 1 | P_{ij} = 0) \ (= 1 - tn), \tag{2.4}$$

$$fn = \Pr(O_{ij} = 0 | P_{ij} = 1) \ (= 1 - tp). \tag{2.5}$$

Then, $\Pr(O_{ij} = 1)$ is given by

$$\begin{aligned} &\Pr(O_{ij} = 1) \\ &= \Pr(O_{ij} = 1, P_{ij} = 1) + \Pr(O_{ij} = 1, P_{ij} = 0) \qquad (2.6) \\ &= \Pr(P_{ij} = 1)(1 - fn) + (1 - \Pr(P_{ij} = 1))fp. \qquad (2.7) \end{aligned}$$

Deng et al. [14] defined the likelihood function (the probability of the observed whole proteome interaction data) by

$$L = \prod_{o_{ij} \in \mathcal{O}} \Pr(O_{ij} = o_{ij}), \tag{2.8}$$

where $\mathcal{O}$ is a set of observations between proteins, and $o_{ij} = 1$ if the interaction between P_i and P_j is observed. Otherwise, $o_{ij} = 0$.

The likelihood L is a function taking λ_{mn}, fp and fn as its arguments. Since it is difficult to directly compute λ_{mn}, fp and fn which maximize L, Deng et al. applyed an EM algorithm [13], where fp and fn were fixed to certain values from biological experimental results.

We briefly review the EM algorithm to confirm their equation because they have not shown details of the deriving process. Let $\mathcal{D}^{ev}$ be an event of interactions between all domains as follows,

$$\mathcal{D}^{ev} = \bigcup_{D_{mn} \in \mathcal{O}} \{D_{mn} = 0 \text{ or } 1\}, \tag{2.9}$$

where both of $D_{mn} = 1$ and $D_{mn} = 0$ for the same domain pair (D_m, D_n) are not included in any $\mathcal{D}^{ev}$ simultaneously, and $D_{mn} \in \mathcal{O}$ means that the domain pair D_{mn} is contained in protein pairs included in $\mathcal{O}$.

For the probability that proteins P_i and P_j interact, as the hidden variables, it is sufficient to consider only probabilities with domain pairs (D_m, D_n) which are included in the protein pair (P_i, P_j). Let $\mathcal{D}^{ev}_{ij}$ be the set which is

$\mathcal{D}^{ev}$ restricted to their domain pairs. The joint probability that observed data are really observed and a domain condition of $\mathcal{D}_{ij}^{ev}$ occurs is

$$
\begin{aligned}
P(O_{ij} = o_{ij}, \mathcal{D}_{ij}^{ev}) &= P(o_{ij}, \mathcal{D}_{ij}^{ev}) && (2.10)\\
&= P(\mathcal{D}_{ij}^{ev}) P(o_{ij} | \mathcal{D}_{ij}^{ev}) \\
&= \prod_{D_{mn} \in P_{ij}} (\lambda_{mn})^{C(\lambda_{mn}, \mathcal{D}_{ij}^{ev})} (\overline{\lambda}_{mn})^{C(\overline{\lambda}_{mn}, \mathcal{D}_{ij}^{ev})} P(o_{ij} | \mathcal{D}_{ij}^{ev}), && (2.11)
\end{aligned}
$$

where $\overline{\lambda}_{mn} = P(D_{mn} = 0) = 1 - \lambda_{mn}$, and $C(\lambda_{mn}, \mathcal{D}_{ij}^{ev})$ $(C(\overline{\lambda}_{mn}, \mathcal{D}_{ij}^{ev}))$ is the degree of λ_{mn} $(\overline{\lambda}_{mn})$ on a probability formula $P(\mathcal{D}_{ij}^{ev})$ under the condition $\mathcal{D}_{ij}^{ev}$. If the event that $D_{mn} = 1$ appears in $\mathcal{D}_{ij}^{ev}$, $C(\lambda_{mn}, \mathcal{D}_{ij}^{ev}) = 1$ and $C(\overline{\lambda}_{mn}, \mathcal{D}_{ij}^{ev}) = 0$. Otherwise, that is if $D_{mn} = 0$, then $C(\lambda_{mn}, \mathcal{D}_{ij}^{ev}) = 0$ and $C(\overline{\lambda}_{mn}, \mathcal{D}_{ij}^{ev}) = 1$.

According to Dempster et al. [13], in order to maximize the likelihood function L, it is sufficient to maximize the following function Q defined as a conditional expectation of the likelihood L,

$$
\begin{aligned}
Q(\theta; \theta') &= E_{\{\theta' | \mathcal{O}\}}(\ln L(\theta)) && (2.12)\\
&= \sum_{o_{ij} \in \mathcal{O}} \sum_{\mathcal{D}_{ij}^{ev}} P(\mathcal{D}_{ij}^{ev} | o_{ij}; \theta') \ln P(o_{ij}, \mathcal{D}_{ij}^{ev}; \theta), && (2.13)
\end{aligned}
$$

where $\theta = \{\lambda_{mn}, \overline{\lambda}_{mn}\}$ is a set of variables, $\theta' = \{\lambda'_{mn}, \overline{\lambda}'_{mn}\}$ is a set of constant values, the semi-colon of $P(x; \theta)$ means that the function $P(x)$ uses the parameters θ, and 'ln' denotes the natural logarithm.

In order to add the constraint $\lambda_{mn} + \overline{\lambda}_{mn} = 1$, we use a Lagrange multiplier κ_{mn}, and the function is written as follows:

$$
\begin{aligned}
&\mathcal{L}(\lambda_{mn}, \overline{\lambda}_{mn}) \\
&= Q(\theta; \theta') + \sum_{D_{mn} \in \mathcal{O}} \kappa_{mn} (1 - \lambda_{mn} - \overline{\lambda}_{mn}) && (2.14)\\
&= \sum_{o_{ij} \in \mathcal{O}} \sum_{\mathcal{D}_{ij}^{ev}} P(\mathcal{D}_{ij}^{ev} | o_{ij}; \theta') \ln P(o_{ij}, \mathcal{D}_{ij}^{ev}; \theta) \\
&\quad + \sum_{D_{mn} \in \mathcal{O}} \kappa_{mn} (1 - \lambda_{mn} - \overline{\lambda}_{mn}) && (2.15)\\
&= \sum_{o_{ij} \in \mathcal{O}} \sum_{\mathcal{D}_{ij}^{ev}} P(\mathcal{D}_{ij}^{ev} | o_{ij}; \theta') \\
&\quad \times \sum_{D_{mn} \in P_{ij}} \Big(C(\lambda_{mn}, \mathcal{D}_{ij}^{ev}) \ln \lambda_{mn}
\end{aligned}
$$

$$+ C(\overline{\lambda}_{mn}, \mathcal{D}_{ij}^{ev}) \ln \overline{\lambda}_{mn} + \ln P(o_{ij}|\mathcal{D}_{ij}^{ev})\Big) + \sum_{D_{mn}\in\mathcal{O}} \kappa_{mn}(1 - \lambda_{mn} - \overline{\lambda}_{mn}). \quad (2.16)$$

Differentiating this function with respect to each variable, we have

$$\frac{\partial \mathcal{L}}{\partial \lambda_{mn}} = \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \sum_{\mathcal{D}_{ij}^{ev}} P(\mathcal{D}_{ij}^{ev}|o_{ij};\theta') \frac{C(\lambda_{mn}, \mathcal{D}_{ij}^{ev})}{\lambda_{mn}} - \kappa_{mn}, \quad (2.17)$$

$$\frac{\partial \mathcal{L}}{\partial \overline{\lambda}_{mn}} = \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \sum_{\mathcal{D}_{ij}^{ev}} P(\mathcal{D}_{ij}^{ev}|o_{ij};\theta') \frac{C(\overline{\lambda}_{mn}, \mathcal{D}_{ij}^{ev})}{\overline{\lambda}_{mn}} - \kappa_{mn}. \quad (2.18)$$

If $\partial\mathcal{L}/\overline{\lambda}_{mn}$ and $\partial\mathcal{L}/\overline{\lambda}_{mn}$ are set to 0 to maximize $Q(\theta;\theta')$, we have

$$\lambda_{mn} = \frac{1}{\kappa_{mn}} \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \sum_{\mathcal{D}_{ij}^{ev}} P(\mathcal{D}_{ij}^{ev}|o_{ij};\theta') C(\lambda_{mn}, \mathcal{D}_{ij}^{ev}) \quad (2.19)$$

$$= \frac{1}{\kappa_{mn}} \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \sum_{\mathcal{D}_{ij}^{ev}} C(\lambda_{mn}, \mathcal{D}_{ij}^{ev}) \frac{P(o_{ij}, \mathcal{D}_{ij}^{ev};\theta')}{P(o_{ij};\theta')}. \quad (2.20)$$

Since $C(\lambda_{mn}, \mathcal{D}_{ij}^{ev}) = 1$ if and only if $D_{mn} = 1$ appears in $\mathcal{D}_{ij}^{ev}$, we have

$$\lambda_{mn} = \frac{1}{\kappa_{mn}} \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \sum_{\{\mathcal{D}_{ij}^{ev}|D_{mn}=1\}} \frac{P(o_{ij}, \mathcal{D}_{ij}^{ev};\theta')}{P(o_{ij};\theta')}. \quad (2.21)$$

Because events of domain-domain interactions are independent from each other, and $P(D_{mn} = 1;\theta') = \lambda'_{mn}$, we have

$$\lambda_{mn} = \frac{1}{\kappa_{mn}} \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \sum_{\{\mathcal{D}_{ij}^{ev}|D_{mn}=1\}} \frac{\lambda'_{mn} P(\mathcal{D}_{ij}^{ev}/D_{mn};\theta') P(o_{ij}|\mathcal{D}_{ij}^{ev};\theta')}{P(o_{ij};\theta')}, \quad (2.22)$$

where $\mathcal{D}_{ij}^{ev}/D_{mn}$ means the set $\mathcal{D}_{ij}^{ev}$ without the event of D_{mn}.

Since the above $\mathcal{D}_{ij}^{ev}$ includes the event $D_{mn} = 1$, which also means $P_{ij} = 1$, if $o_{ij} = 1$, $P(o_{ij}|\mathcal{D}_{ij}^{ev}) = tp$, otherwise $P(o_{ij}|\mathcal{D}_{ij}^{ev}) = fp$. Substituting them, we have

$$\lambda_{mn} = \frac{\lambda'_{mn}}{\kappa_{mn}} \Bigg(\sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\wedge o_{ij}=1\}} \sum_{\{\mathcal{D}_{ij}^{ev}|D_{mn}=1\}} \frac{P(\mathcal{D}_{ij}^{ev}/D_{mn};\theta') tp}{P(o_{ij};\theta')}$$

$$+ \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\wedge o_{ij}=0\}} \sum_{\{\mathcal{D}_{ij}^{ev}|D_{mn}=1\}} \frac{P(\mathcal{D}_{ij}^{ev}/D_{mn};\theta')fn}{P(o_{ij};\theta')}\Bigg) \tag{2.23}$$

$$= \frac{\lambda'_{mn}}{\kappa_{mn}}\Bigg(\sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\wedge o_{ij}=1\}} \frac{tp}{P(o_{ij};\theta')} + \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\wedge o_{ij}=0\}} \frac{fn}{P(o_{ij};\theta')}\Bigg) \tag{2.24}$$

$$= \frac{\lambda'_{mn}}{\kappa_{mn}} \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \frac{(1-fn)^{o_{ij}} fn^{1-o_{ij}}}{P(o_{ij};\theta')}. \tag{2.25}$$

In a similar way, we can calculate $\overline{\lambda}_{mn}$ from Equation (2.21) as follows,

$$\overline{\lambda}_{mn} = \frac{1}{\kappa_{mn}} \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \sum_{\{\mathcal{D}_{ij}^{ev}|D_{mn}=0\}} \frac{P(o_{ij},\mathcal{D}_{ij}^{ev};\theta')}{P(o_{ij};\theta')}. \tag{2.26}$$

Taking into account the constraint of probabilities,

$$\lambda_{mn} + \overline{\lambda}_{mn} = \frac{1}{\kappa_{mn}} \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \sum_{\mathcal{D}_{ij}^{ev}} \frac{P(o_{ij},\mathcal{D}_{ij}^{ev};\theta')}{P(o_{ij};\theta')} \tag{2.27}$$

$$= \frac{1}{\kappa_{mn}} \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} 1 = 1. \tag{2.28}$$

Therefore,

$$\kappa_{mn} = N_{mn} \tag{2.29}$$

Consequently, we have an equation to update λ_{mn}:

$$\lambda_{mn} = \frac{\lambda'_{mn}}{N_{mn}} \sum_{\{o_{ij}\in\mathcal{O}|D_{mn}\in P_{ij}\}} \frac{(1-fn)^{o_{ij}} fn^{1-o_{ij}}}{P(o_{ij};\theta')}. \tag{2.30}$$

2.3.3 LPBN: LP-based Method for Binary Interaction Data

In this subsection, I describe the proposed LP-based method for inferring protein-protein interactions.

Using the probabilistic model and a threshold Θ, we can predict protein-protein interactions by the following rule:

$$P_i \text{ and } P_j \text{ interact} \iff 1 - \prod_{D_{mn}\in P_{ij}} (1-\lambda_{mn}) \geq \Theta. \tag{2.31}$$

The condition can be transformed as follows:

$$1 - \prod_{D_{mn} \in P_{ij}} (1 - \lambda_{mn}) \geq \Theta, \tag{2.32}$$

$$\prod_{D_{mn} \in P_{ij}} (1 - \lambda_{mn}) \leq 1 - \Theta, \tag{2.33}$$

$$\ln \left(\prod_{D_{mn} \in P_{ij}} (1 - \lambda_{mn}) \right) \leq \ln(1 - \Theta), \tag{2.34}$$

$$\sum_{D_{mn} \in P_{ij}} \ln(1 - \lambda_{mn}) \leq \ln(1 - \Theta). \tag{2.35}$$

Let $\gamma_{mn} = \ln(1 - \lambda_{mn})$ and $\beta = \ln(1 - \Theta)$. Then, the above condition can be written as

$$\sum_{D_{mn} \in P_{ij}} \gamma_{mn} \leq \beta. \tag{2.36}$$

This is a linear inequality. Therefore, if we can find γ_{mn} ($\gamma_{mn} \leq 0$) satisfying

$$O_{ij} = 1 \iff \sum_{D_{mn} \in P_{ij}} \gamma_{mn} \leq \beta \tag{2.37}$$

for all observed data (i.e., all training data) O_{ij}, we can obtain the necessary parameters consistent with all training data.

However, it is usually impossible to satisfy all constraints. In such a case, it is reasonable to try to minimize the classification error. Though it is quite difficult to minimize the number of unsatisfied constraints [1], it is possible to minimize the sum of distances [8, 12]. Therefore, we use the following linear program:

$$\begin{aligned}
\text{minimize} \quad & \sum_{o_{ij} \in \mathcal{O}} \xi_{ij}, \\
\text{subject to} \quad & \sum_{D_{mn} \in P_{ij}} \gamma_{mn} \leq \beta - const + \xi_{ij} \\
& \qquad \text{for } P_{ij} \text{ such that } O_{ij} = 1, \\
& \sum_{D_{mn} \in P_{ij}} \gamma_{mn} > \beta + const - \xi_{ij} \\
& \qquad \text{for } P_{ij} \text{ such that } O_{ij} = 0, \\
& \gamma_{mn} \leq 0 \quad \text{for all } \gamma_{mn}, \\
& \xi_{ij} \geq 0 \quad \text{for all } \xi_{ij}, \\
& \beta < 0,
\end{aligned}$$

where *const* is an appropriate small constant (we currently use $const = 0.01$). Once γ_{mn} and β are determined, we can obtain λ_{mn} and Θ by $\lambda_{mn} = 1 - \exp(\gamma_{mn})$ and $\Theta = 1 - \exp(\beta)$, respectively.

2.3.4 Combination of LPBN and EM

Due to the relation of $\lambda_{mn} = 1 - \exp(\gamma_{mn})$ (equivalently, $\gamma_{mn} = \ln(1 - \lambda_{mn})$), we can combine the LPBN method with the EM method. We examine two kinds of combinations: LPEM and EMLP.

The LPEM method first computes γ_{mn} using LPBN. Then, it converts γ_{mn} into λ_{mn} and applies the EM method using these λ_{mn} as the initial values.

The EMLP method first computes λ_{mn} using the EM method. Next, the following constraints are added to the linear program:

$$\ln((1-\delta)(1-\lambda_{mn})) \leq \gamma_{mn} \leq \ln((1+\delta)(1-\lambda_{mn})), \quad (2.38)$$

where δ is an appropriate fixed constant (we currently use $\delta = 0.05$ and $\delta = 0.2$). Then, γ_{mn} are obtained by solving the linear program.

2.3.5 SVM-based Method

It is reasonable to apply SVM to inference of protein-protein interactions because LPBN is similar to SVM [12]. Although SVM was already applied to inference of protein-protein interactions by Bock et al. [10], they did not compute scores or probabilities of domain-domain interactions. In order to apply SVM to inference of domain-domain interactions, we treat observed interacting pairs as positive examples and non-observed pairs as negative examples. For each protein pair (P_i, P_j), we define the feature vector $\boldsymbol{f}_{ij}$ by

$$\boldsymbol{f}_{ij}^{(mn)} = \begin{cases} 1 & \text{if } D_{mn} \in P_{ij} \\ 0 & \text{otherwise,} \end{cases} \quad (2.39)$$

where $\boldsymbol{f}_{ij}^{(mn)}$ denotes the mn-th element of the vector $\boldsymbol{f}_{ij}$. If we apply the linear kernel and the soft margin to the SVM, it will be quite similar to LPBN. But, there is a big difference. In the SVM formulation, we can not guarantee

$\gamma_{mn} \leq 0$ (recall that $\gamma_{mn} = \ln(1 - \lambda_{mn})$). This condition is very important to give the probabilistic interpretation for the obtained parameters.

2.4 Data and Implementation

I compared the LP-based methods (LPBN, LPEM and EMLP) with the association method (ASSOC) and the EM method (EM). For the training and test data of protein-protein interactions, I used the full data set (Scere20040404.tab) and the core data sets (core20020404.lst and ScereCR-20040404.tab) of *Saccharomyces cerevisiae* from the DIP database [44]. The core data sets were more reliable than the full data set. The DIP database seems to consist of the most reliable interaction data. For each protein in this database, I obtained its sequence data from the Swissprot/TrEMBL database [4]. In order to derive domains from the sequences, I used InterProScan (version 3.1) [45] as in [27, 37]. Though InterProScan identified not only protein domains but also protein signatures such as functional sites and sequence motifs, I used all the hits because signatures may also play an important role in protein-protein interaction. As in [27, 37], InterPro signatures in the same parent-child relationship were also merged into one signature. The sequence and signature pairs I used can be found at http://sunflower.kuicr.kyoto-u.ac.jp/~morihiro/protint/supplement.html.

I used SVM^{light} [26] for SVM learning, and used LOQO (version 1.08) on SUN UNIX [39] for solving linear programs. The experiments were mostly performed on a PC cluster with 8 Pentium Xeon 2.8 GHz processors, where only one CPU was used in all experiments. In each case, both training and tests could be done in a few minutes.

The scores obtained by ASSOC were used as the initial values of λ_{mn} for EM since it was much better to use these scores than to use random initial values. EM steps were repeated until the difference of log-likelihood between two consecutive steps became less than 0.01 or until the number of repeats exceeded 200. Following to [14], $fp = 2.5 \times 10^{-4}$ and $fn = 0.80$ were used for EM. Though I examined several other parameter sets for EM, the results did not change significantly. I used the linear kernel for SVM with the default value of the trade-off parameter. Though I examined other kernels

and parameters, the results did not change significantly.

I evaluated the methods using the rate of correct answers and the relationship between sensitivity and specificity. We call a protein pair a *true positive* if it is both predicted and observed, a *false positive* if it is predicted but is not observed, a *true negative* if it is neither predicted nor observed, and a *false negative* if it is not predicted but is observed. The *rate of correct answers* is defined to be the ratio of the number of true positives plus true negatives to the total number of examples. The *sensitivity* is defined to be the ratio of the number of true positives to the number of true positives plus false negatives. The *specificity* is defined to be the ratio of the number of true negatives to the number of true negatives plus false positives.

In order to classify predicted probabilities of protein-protein interactions for test data, a threshold is required. Although the LPBN method estimates the threshold with other parameters simultaneously, the association and EM methods do not estimate it. Therefore, for the LPBN method, I used the estimated value ($\Theta = 1 - \exp(\beta)$) as the threshold, and for other methods, I used the value to maximize the rate of correct answers for a training data set.

2.5 Results

I first used the DIP data sets (ScereCR20040404.tab (core data set) and Scere20040404.tab (full data set)). Among 5552 pairs in the DIP core data set, I used 4533 pairs as positive data (POS), for each of which at least one hit was found by InterProScan. In this data set, the number of domains was 1222 and the number of proteins was 1863. For the full data set, I used 9159 protein pairs as POS among 12205 pairs, where the number of domains was 1697 and the number of proteins was 2840.

I compared the methods using a standard evaluation procedure: parameters were learned using the training data set and then the rate of correct answers, the sensitivity and the specificity were measured using the test data set. I randomly selected protein pairs not contained in POS as negative training data. In addition, I randomly selected 2/3 of POS and NEG as training data and the remaining 1/3 of POS and NEG as test data.

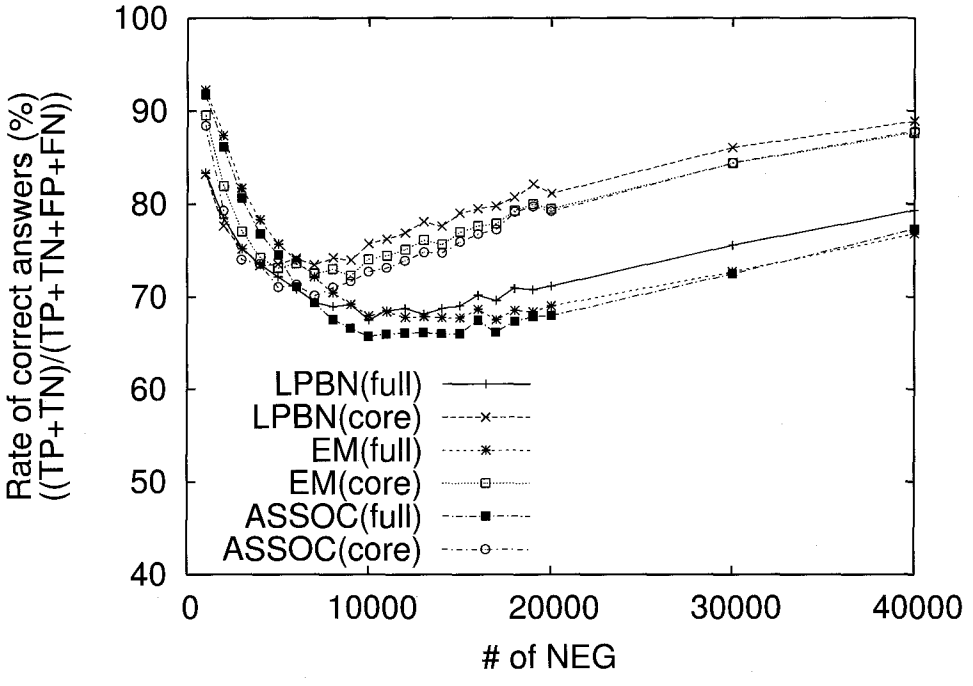

Figure 2.2: The rate of correct answers of LPBN, EM and ASSOC for both core and full data sets. The horizontal axis indicates the amount of negative data added (NEG). It is seen that LPBN was the best for the both core and full data sets.

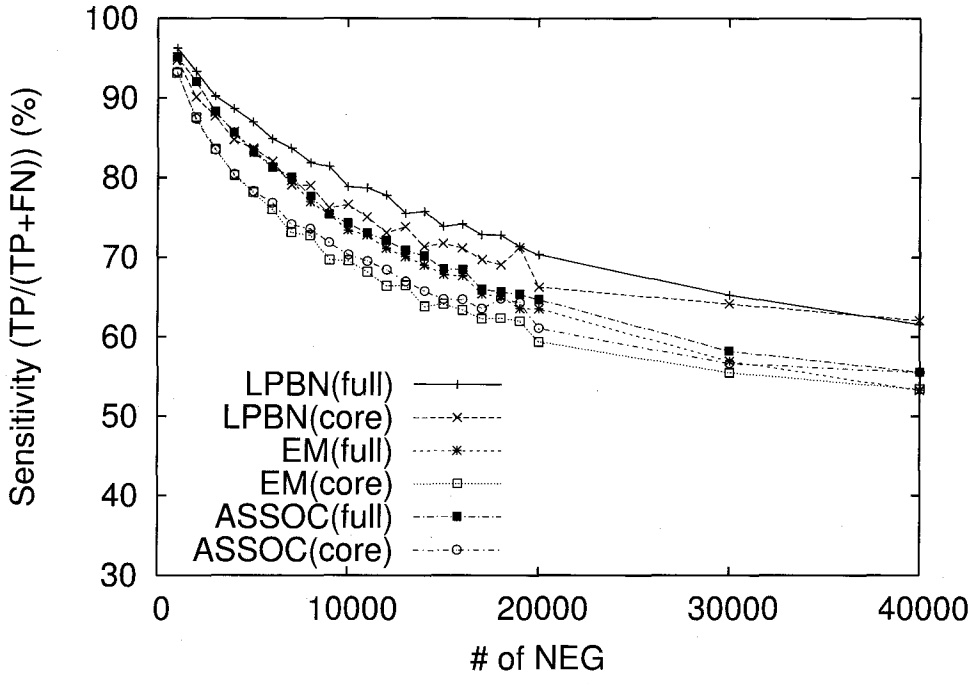

Figure 2.3: The sensitivity of LPBN, EM and ASSOC for both core and full data sets. The horizontal axis indicates the amount of negative data added (NEG). It is seen that LPBN was the best for the both core and full data sets.

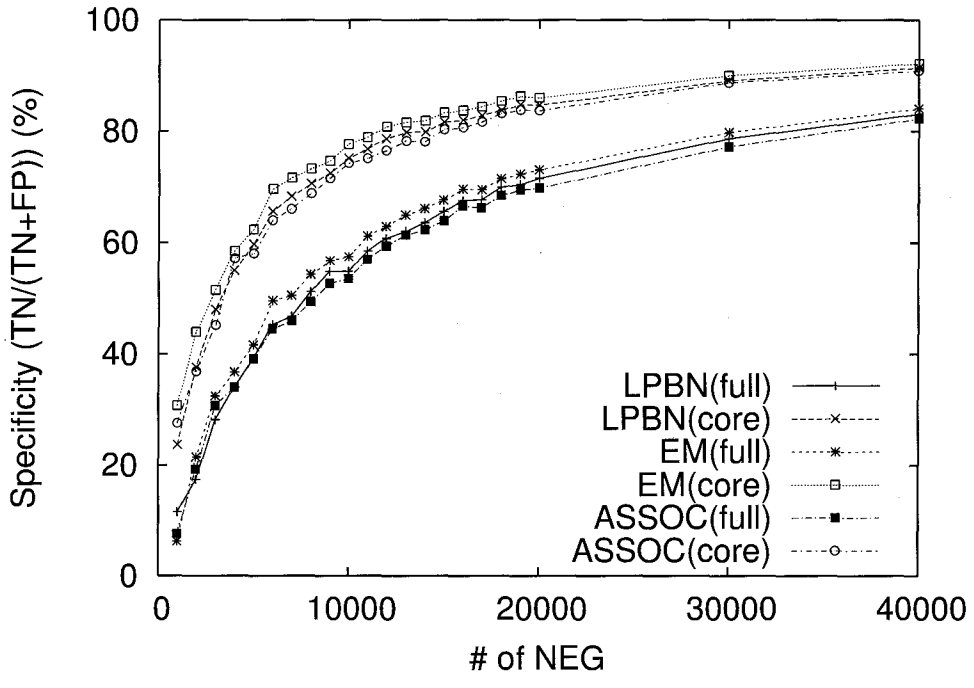

Figure 2.4: The specificity of LPBN, EM and ASSOC for both core and full data sets. The horizontal axis indicates the amount of negative data added (NEG). It is seen that they were comparable. When NEG was small, the resulting specificity was low due to the bias of training data.

The result is shown in Figures 2.2, 2.3 and 2.4. Figures 2.2, 2.3 and 2.4 show the rate of correct answers, the sensitivity and the specificity of LPBN, EM and ASSOC for the both core and full data sets, respectively.

In Figure 2.2, it is seen that the rate of correct answers of LPBN was almost always better than other methods for the both data sets. In Figure 2.3, it is seen that the sensitivity of LPBN was also the best. In Figure 2.4 it is seen that the specificities of the methods were comparable. On predicting protein-protein interactions, the sensitivity is more important than the specificity because the sensitivity is the ability to truly discriminate interacting protein pairs. When NEG is around 0, the sensitivities are high, and the specificities are low due to the bias of training data.

Next, I examined the relationship between sensitivity and specificity. I used the DIP core data set (core20020404.lst). Among 3003 pairs in the DIP core data set, I used 1767 pairs as positive data (POS), for each of which at least one hit was found by InterProScan. The other protein pairs were used as negative data (NEG), where I only considered the proteins that appeared in POS. Because of the limit of memory space, only (randomly selected) 40% of NEG were given for LPBN and SVM. Parameters were learned using the training data set and then the sensitivity and the specificity were calculated. I randomly selected 2/3 of POS as positive training data, and the remaining 1/3 of POS as positive test data. I randomly selected protein pairs not contained in POS as negative training data. I repeated the above procedure 10 times and took the average over 10 trials.

The result is shown in Figure 2.5. Since performances of LPEM and EMLP were almost the same as EM, the curve for LPEM or EMLP is not drawn in Figure 2.5. It is seen that LPBN, EM and ASSOC were comparable, and SVM is poor. It is suggested from the figure that the probabilistic model proposed by Deng et al. [14] is appropriate because SVM is not based on the model whereas the other methods are based on the model.

The relationship between sensitivity and specificity for the test data set is shown in Figure 2.6. It should be noted that I removed protein pairs in the test data set which did not have domain pairs appearing in the positive training data set because the scores of such pairs are always 0. If such pairs are included, the sensitivity will decrease significantly. For example, the

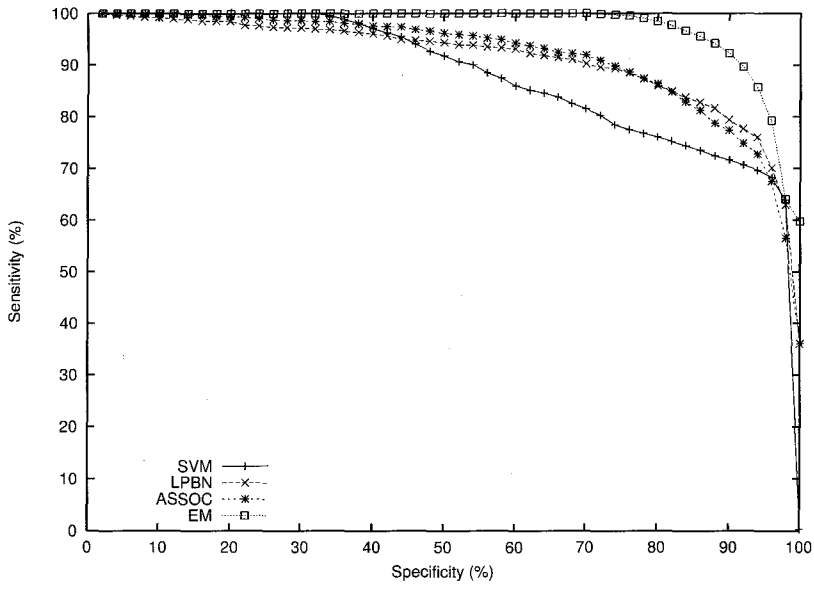

Figure 2.5: Comparison of specificity and sensitivity for several methods on training data. It is seen that EM is the best, LPBN and ASSOC are comparable, and SVM is poor.

sensitivity decreases to $50 \sim 60\%$ when specificity=80% in each method.

It is seen from Figure 2.6 that the performance of SVM was poor. As in the case of training data, the performance of LPEM was similar to that of EM. Since the differences among EMLP, ASSOC and EM were unclear from Figure 2.6, the details of a part of Figure 2.6 are shown in Figure 2.7 for these three methods. It is seen that EMLP was slightly better than EM, and EM was slightly better than ASSOC. Though EM was better than EMLP in the region of specificity $< 50\%$, the region of specificity $\geq 50\%$ is much more important because the threshold to classify interactions is determined in this region.

In Figure 2.5, EM was better than others for the training data set. However, in Figure 2.6, the differences for the test data set were small. In fact, EM was worse than ASSOC for several cases in which a lot of negative training data were given. It is probably due to overfitting. Thus, we might be able to improve the prediction accuracy for the test data set if some technique for avoiding overfit can be incorporated into EM and/or the LP-based method.

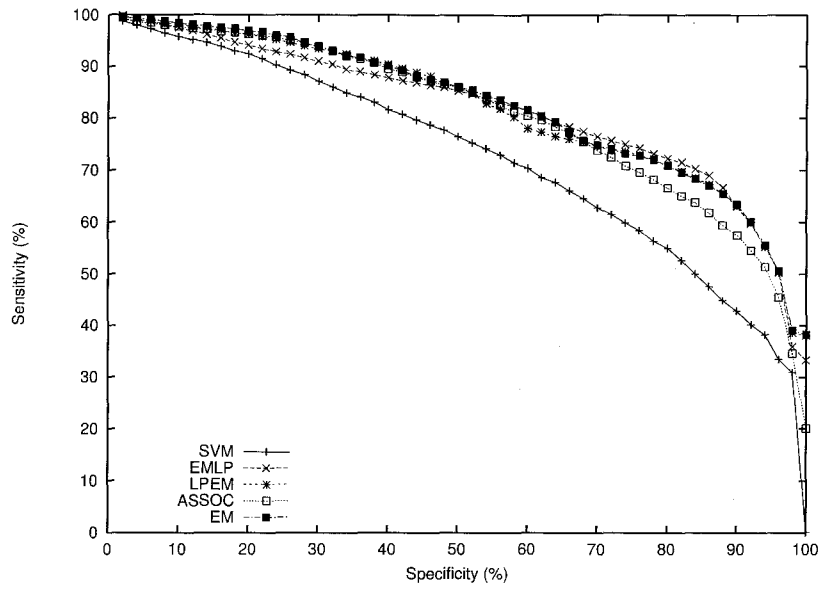

Figure 2.6: Comparison of specificity and sensitivity for SVM, EMLP, LPEM, ASSOC and EM on test data.

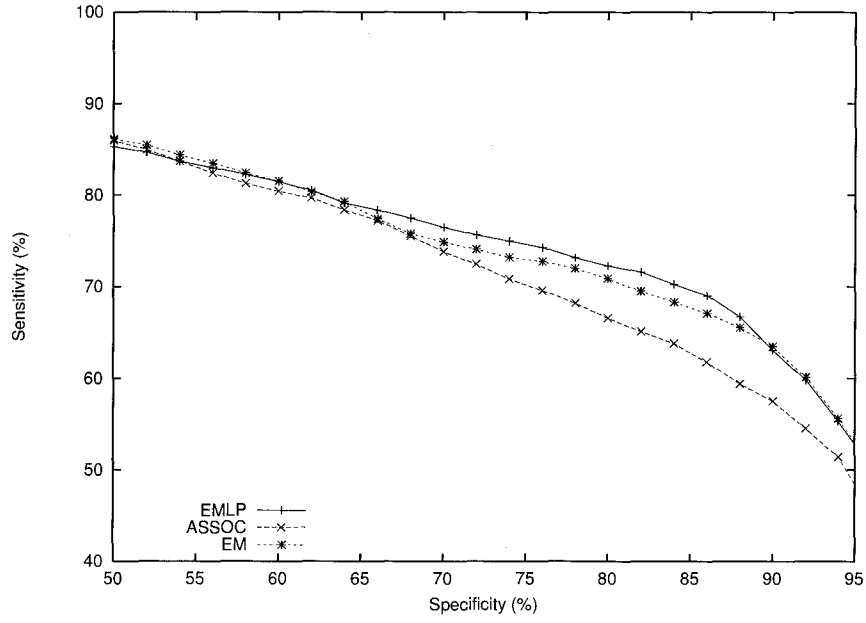

Figure 2.7: Detailed comparison of specificity and sensitivity for EMLP, ASSOC and EM on test data.

2.6 Discussion

I proposed an LP-based method (along with several variants) for inferring protein-protein interactions from experimental data. I compared the proposed method with existing methods such as the association method and the EM method. It is seen that the rate of correct answers and the sensitivity of LPBN are better than other methods.

The proposed method has the feature that several kinds of constraints can be added. In this chapter, I used constraints on the ranges of the parameter values (EMLP). It was useful to combine the LP-based method with the EM method. It would be interesting to seek other types of constraints.

As mentioned before, all examined methods except the SVM-based method are based on the probabilistic model proposed by Deng et al. [14] and are better than the SVM-based method. This suggests that the probabilistic model by Deng et al. [14] is adequate and might capture some features of the relationship between domain-domain interactions and protein-protein interactions.

Though the LPBN method was better than the SVM-based method, it is similar to the SVM-based method in the sense that both methods use a hyperplane to separate positive examples from negative examples, and try to minimize the sum of classification errors. If SVM can be modified for cooperating with constraints that the parameters must be negative, better results might be obtained. It would be interesting to study such modifications since SVMs have been successfully applied to many problems in Bioinformatics. It would also be interesting to modify SVM so that it can cooperate with numerical training data which I describe in chapter 4.

Chapter 3

Hardness of Inferring Protein-Protein Interactions

In the previous chapter, I proposed a practical method of inferring protein-protein interactions. In this chapter, we consider to derive new algorithms which are more accurate and more efficient for inferring interactions. From the point of view of time complexity, I will show that an inference problem for interaction data is contained in the class of MAX SNP-hard. This result means that it is inherently intractable to infer interactions. Under the assumption of P≠NP, we can not obtain an optimal solution for the problem in polynomial time. Moreover, we can not construct any polynomial-time algorithm which guarantees an arbitrary approximation.

3.1 Problem Definition of Protein-Protein Interactions

First, we formulate the problem of inferring protein-protein interactions as a maximization problem. Following the definition of the original model [14], we introduce a parameter Θ as a threshold for predicting protein-protein interactions. With Θ, we predict protein-protein interactions by the following rule:

$$P_i \text{ and } P_j \text{ interact} \Leftrightarrow \Pr(P_{ij}=1) = 1 - \prod_{D_{mn} \in P_{ij}} (1-\lambda_{mn}) \geq \Theta \quad (3.1)$$

$$\Leftrightarrow \prod_{D_{mn} \in P_{ij}} (1 - \lambda_{mn}) \leq 1 - \Theta. \tag{3.2}$$

Then, we define a maximization problem (MAX PPI (MAXimization problem of Protein-Protein Interactions)) as follows,

Problem 1 (MAX PPI) *Let $\mathcal{P}_{\rm pos}$ and $\mathcal{P}_{\rm neg}$ be a multi set of interacting protein pairs (P_i, P_j) and a multi set of non-interacting protein pairs, respectively. Let λ_{mn} denote the probability that domains D_m and D_n interact. We consider two types of inequalities,*

$$\begin{aligned} &\prod_{D_{mn} \in P_{ij}} (1 - \lambda_{mn}) \leq 1 - \Theta \quad \textit{if a protein pair } (P_i, P_j) \textit{ is in } \mathcal{P}_{\rm pos}, \\ &\prod_{D_{mn} \in P_{ij}} (1 - \lambda_{mn}) > 1 - \Theta \quad \textit{if a protein pair } (P_i, P_j) \textit{ is in } \mathcal{P}_{\rm neg}. \end{aligned} \tag{3.3}$$

Given $\mathcal{P}_{\rm pos}$, $\mathcal{P}_{\rm neg}$, and sets $P_1, \ldots, P_N$ of domains, find the parameters λ_{mn} and Θ to maximize the number of the inequalities that are satisfied.

Note that $\mathcal{P}_{\rm pos}, \mathcal{P}_{\rm neg}$ are multi sets because there can be several proteins which have the same composition of domains. However, it seems possible to prove the following Theorem (1) even if we assume that $\mathcal{P}_{\rm pos}, \mathcal{P}_{\rm neg}$ are general sets.

Theorem 1 *MAX PPI is in the class of MAX SNP-hard.*

In this chapter, I will show this theorem for multi sets of $\mathcal{P}_{\rm pos}, \mathcal{P}_{\rm neg}$.

3.2 Review of MAX SNP-hard

Here, we review the class of MAX SNP-hard. MAX SNP [35] is a class of optimization (maximization or minimization) problems in the class SNP (Strict NP), which is a subclass of NP. Each problem of SNP can be written as a predicate logic formula,

$$\exists S \forall x \psi(x, G, S), \tag{3.4}$$

where ψ, S and G are predicates, ψ is first order and quantifier free, S is a second order predicate variable and is restricted by an existential quantifier,

G is quantifier free, and x is a variable. We can represent a maximization problem of MAX SNP from this expression as follows:

$$\max_S\{x|\psi(x,G,S)\}. \tag{3.5}$$

For example, there is MAX 3SAT [35] which is a problem in MAX SNP. MAX 3SAT is a maximization problem of 3SAT. 3SAT is one of the satisfiability problems, and each clause of the instance has three literals. In other words, 3SAT discriminates whether the number of satisfied clauses is exactly the number of all of the clauses input or not. On the other hand, MAX 3SAT maximizes the number of satisfied clauses. It is written in terms of predicate logic formulas as follows,

$$\begin{aligned}\max_T|\{(x_1,x_2,x_3)|&(C_0(x_1,x_2,x_3)\Rightarrow x_1\in T\vee x_2\in T\vee x_3\in T)\\ &\wedge(C_1(x_1,x_2,x_3)\Rightarrow x_1\notin T\vee x_2\in T\vee x_3\in T)\\ &\wedge(C_2(x_1,x_2,x_3)\Rightarrow x_1\notin T\vee x_2\notin T\vee x_3\in T)\\ &\wedge(C_3(x_1,x_2,x_3)\Rightarrow x_1\notin T\vee x_2\notin T\vee x_3\notin T)\}|,\end{aligned} \tag{3.6}$$

where $C_j(x_1,x_2,x_3)$ is a predicate and means that there is a clause in which by sorting three variables x_1,x_2,x_3, j variables $x_1,\cdots,x_j$ appear as negative literals with the remaining $x_{j+1},\cdots,x_3$ appearing as positive literals, and T is a second order predicate variable and means a set of variables x_i assigned to true. Note that $A\Rightarrow B$ is logically equivalent to $\neg A\vee B$ for variables A,B.

T corresponds to S in the expressions (3.4) and (3.5). The C_js correspond to G.

If a problem can be reduced from the class problems that are in MAX SNP by L-reduction [35], the problem is in MAX SNP-hard, which is simultaneously in a subclass of NP-hard. In addition, if the problem is in MAX SNP, it is in MAX SNP-complete.

The L-reduction is a very restricted form of transformation, and is defined for treating approximability issues as follows, [35] (see Figure 3.1),

Definition 1 (L-reduction) *Let Π and Π' be two optimization problems. We say that Π L-reduces to Π' if there are two polynomial-time algorithms f, g, and constants $\alpha,\beta>0$ such that for each instance I of Π:*

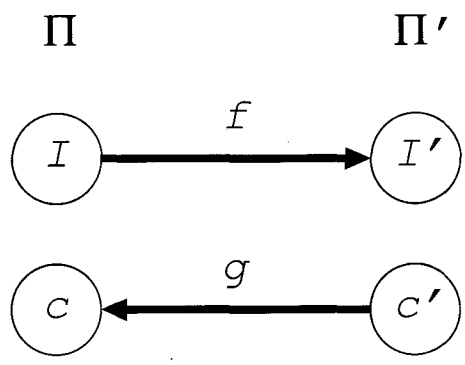

Figure 3.1: L-reduction from Π to Π'. I, I' are instances of Π, Π', respectively, and c, c' are costs of solutions of I, I', respectively. f, g are polynomial-time algorithms. f transforms I to I', and g transforms any solution with c' to a solution with c.

(a) *Algorithm f produces an instance $I' = f(I)$ of Π', such that the optima of I and I', denoted by $OPT(I)$ and $OPT(I')$, respectively, satisfy $OPT(I') \leq \alpha OPT(I)$.*

(b) *Given any solution of I' with cost c', algorithm g produces a solution of I with cost c such that $|c - OPT(I)| \leq \beta |c' - OPT(I')|$.*

Note that a cost c is directly obtained from a solution of I, and $c \leq OPT(I)$ if the Π is one of maximization problems. The constant β will usually be 1.

From this definition, we can obtain the proposition that if Π L-reduces to Π', and there is a polynomial-time approximation algorithm for Π' with worst-case error ϵ, then there is a polynomial-time approximation algorithm for Π with worst-case error $\alpha\beta\epsilon$.

3.3 Proof Overview of Hardness for MAX PPI

For proving Theorem 1, it is sufficient to show that there is an L-reduction from one of the MAX SNP-complete problems to MAX PPI [35]. We use MAX 2UNSAT-B as the MAX SNP-complete problem. From an analogy with MAX 2SAT-B, I define MAX 2UNSAT-B as follows,

Problem 2 (MAX 2UNSAT-B) *Consider a set I of m clauses of $C_1, \cdots, C_m$. Each clause contains up to two literals over a set of Boolean variables $x_1, \cdots, x_n$. Each variable appears at most B times over all clauses of I. B is a constant.*

Given a set I of m clauses, find a truth assignment that maximizes the number of clauses evaluated false.

Problem 3 (MAX 2SAT-B) *Consider a set I of m clauses of $C_1, \cdots, C_m$. Each clause contains up to two literals over a set of Boolean variables $x_1, \cdots, x_n$. Each variable appears at most B times over all clauses of I. B is a constant.*

Given a set I of m clauses, find a truth assignment that maximizes the number of clauses evaluated true.

At first, we have to prove the following completeness for MAX 2UNSAT-B to use it for proving hardness of MAX PPI in Theorem 1:

Theorem 2 *MAX 2UNSAT-B is in the class of MAX SNP-complete.*

3.4 Proof of Completeness for MAX 2UNSAT-B

For proving Theorem 2, it is necessary to show that there is an L-reduction from MAX 2SAT-B which is known to be MAX SNP-complete, and that MAX 2UNSAT-B is in MAX SNP.

Theorem 3 *MAX 2SAT-B is in the class MAX SNP-complete [35].*

Proof First, I will show that there is an L-reduction from MAX 2SAT-B to MAX 2UNSAT-B. I define an algorithm f which transforms a fixed instance I of MAX 2SAT-B to an instance I' of MAX 2UNSAT-B. For each clause $C_i = \{l_{i,1} \vee l_{i,2}\}$ of I, f produces the corresponding clauses C_i' of I' using the set of variables of I as follows:

$$C_i' = \{\bar{l}_{i,1} \vee l_{i,2},\ l_{i,1} \vee \bar{l}_{i,2},\ \bar{l}_{i,1} \vee \bar{l}_{i,2}\}. \tag{3.7}$$

By this transformation, each variable of I' appears at most $3B$ times in the set $\bigcup_i C_i'$ of clauses of I'. I' is an instance of MAX 2UNSAT-B because $3B$ is

a constant. Note that exactly one clause of C'_i is evaluated false if and only if C_i is evaluated true. The algorithm f runs in polynomial time clearly.

Consider the optima $OPT(I)$ and $OPT(I')$. Since one of the truth assignments of $OPT(I)$ is always equivalent to one of the truth assignments of $OPT(I')$, the following equation is satisfied,

$$OPT(I') = OPT(I) \tag{3.8}$$

This equation means that the condition (a) of Definition 1 concerning L-reduction is satisfied by substituting $\alpha = 1$.

In addition, I define the algorithm g to transform solutions from MAX 2UNSAT-B to MAX 2SAT-B so that each variable of I has the same truth assignment as I'. Given a solution of I' $(= f(I))$ with cost c', g produces a solution of I with cost $c = c'$. Therefore, the following equation is satisfied,

$$|c - OPT(I)| = |c' - OPT(I')|. \tag{3.9}$$

This equation means that the condition (b) of Definition 1 is satisfied by substituting $\beta = 1$.

From these, there is an L-reduction from MAX 2SAT-B to MAX 2UNSAT-B, and MAX 2UNSAT-B is in MAX SNP-hard.

Next, I will show that MAX 2UNSAT-B is in MAX SNP. It is sufficient to show that 2UNSAT-B is in SNP. If a problem can be written as the predicate logic formula of Formula 3.4, the problem is in SNP. 2UNSAT-B can be written as follows,

$$\begin{aligned} \exists T \forall (x_1, x_2)((C_0(x_1, x_2) \Rightarrow x_1 \notin T \wedge x_2 \notin T) \\ \wedge (C_1(x_1, x_2) \Rightarrow x_1 \in T \wedge x_2 \notin T) \\ \wedge (C_2(x_1, x_2) \Rightarrow x_1 \in T \wedge x_2 \in T)), \end{aligned} \tag{3.10}$$

where $C_j(x_1, x_2)$ is almost the same as that of the example of MAX 3SAT (see Formula 3.6), which means that there is a clause in which by sorting two variables x_1, x_2, j variables $x_1, \cdots, x_j$ appear as negative literals and the remaining $x_{j+1}, \cdots, x_2$ appear as positive literals, and T means a set of variables x_i assigned to true.

Because this formula is one of Formula 3.4, 2UNSAT-B is in SNP. Therefore, MAX 2UNSAT-B is in MAX SNP, and MAX 2UNSAT-B is in MAX SNP-complete. ∎

3.5 Proof of Hardness for MAX PPI

3.5.1 Preliminary

The following theorem concerning a property of MAX SNP takes a key role in the proof of the hardness of MAX PPI.

Theorem 4 *Every problem in the class of MAX SNP can be approximated in polynomial time within some fixed ratio [35].*

Particularly, since MAX 2UNSAT-B is in the class of MAX SNP, we use the following corollary derived from Theorem 4.

Corollary 1 *Given any instance, I, with m clauses of MAX 2UNSAT-B, there is an algorithm to approximate a solution in polynomial time within the fixed ratio, m/α_r.*

3.5.2 Proof of Hardness for MAX PPI

Proof In order to show there is an L-reduction from MAX 2UNSAT-B to MAX PPI, I will construct algorithms f and g satisfying the conditions of L-reductions.

Consider a fixed instance I of MAX 2UNSAT-B. The instance I consists of m clauses $C_1 \cdots C_m$ with n variables $x_1, \cdots, x_n$. Each clause has exactly two literals, $C_k = \{l_{k,1} \vee l_{k,2}\}$. If we must consider a clause with one literal $C = \{l\}$, we can consider a clause with two literals such as $C' = \{l, l\}$ instead of C. Each literal of $l_{k,1}$ and $l_{k,2}$ is one of variables $x_1, \cdots, x_n$ and their negations $\bar{x}_1, \cdots, \bar{x}_n$.

For this I, by the transformation f, we prepare a set $\mathcal{P}$ of $m + 2Bn + 1$ proteins, and a set $\mathcal{D}$ of $2n + 1$ domains as follows,

$$\mathcal{P} = \{P_1, \cdots, P_{m+2Bn+1}\}, \tag{3.11}$$

$$\mathcal{D} = \{D_{x_1}, D_{\bar{x}_1}, \cdots, D_{x_n}, D_{\bar{x}_n}, D_\alpha\}, \tag{3.12}$$

where α is a new variable, and is not any of the variables of I.

The algorithm f generates a set $\mathcal{P}_{\rm neg}$ of m non-interacting protein pairs for all clauses of I such that each clause C_k of I is corresponding to a protein pair,

$$(\{D_{l_{k,1}}, D_{l_{k,1}}, D_{l_{k,2}}, D_{l_{k,2}}\}, \{D_\alpha\}) \in \mathcal{P}_{\rm neg}. \tag{3.13}$$

Note that in the transformed instance I', as the probability of the interaction between domains $D_{l_{k,1}}$ and D_α for a literal $l_{k,1}$ of C_k of I, a variable $\lambda_{x_i\alpha}$ of I' is used if $l_{k,1} = x_i$, or $\lambda_{\bar{x}_i\alpha}$ is used if $l_{k,1} = \bar{x}_i$. Although we also use $\lambda_{l_{k,1}\alpha}$ in addition to $\lambda_{x_i\alpha}$ and $\lambda_{\bar{x}_i\alpha}$, $\lambda_{l_{k,1}\alpha}$ always means $\lambda_{x_i\alpha}$ or $\lambda_{\bar{x}_i\alpha}$.

I define a Boolean auxiliary variable θ_l for each literal l ($l = x_i$ or $\bar{x}_i$) as follows,

$$\theta_l = \begin{cases} \text{true} & \text{if } 1 - \lambda_{l\alpha} \leq \sqrt{1-\Theta}, \\ \text{false} & \text{otherwise.} \end{cases} \tag{3.14}$$

In $\mathcal{P}_{\rm neg}$, truth values are independently assigned to different Boolean variables θ_{x_i} and $\theta_{\bar{x}_i}$ made from the same variable x_i, and a same truth value can be assigned to both of the variables. In order to avoid this situation, f generates another set, $\mathcal{P}_{\rm pos}$, so that the following conditions are satisfied,

$$\theta_{x_i} = \bar{\theta}_{\bar{x}_i} \text{ for all } i \ (i = 1, \cdots, n). \tag{3.15}$$

For each variable x_i, f generates $2B$ interacting protein pairs as follows,

$$(\{D_{x_i}, D_{\bar{x}_i}\}, \{D_\alpha\}) \in \mathcal{P}_{\rm pos}. \tag{3.16}$$

We summarize the transformed instance I' of MAX PPI.

As the domain compositions of the proteins,

$$P_{2B(i-1)+j} = \{D_{x_i}, D_{\bar{x}_i}\} \quad (1 \leq i \leq n, 1 \leq j \leq 2B), \tag{3.17}$$

$$P_{k+2Bn} = \{D_{l_{k,1}}, D_{l_{k,1}}, D_{l_{k,2}}, D_{l_{k,2}}\} \ (1 \leq k \leq m), \tag{3.18}$$

$$P_{m+2Bn+1} = \{D_\alpha\}. \tag{3.19}$$

As the set $\mathcal{P}_{\rm pos}$ (or $\mathcal{P}_{\rm neg}$) of interacting (or non-interacting) protein pairs,

$$\mathcal{P}_{\rm pos} = \left\{(P_{2B(i-1)+j}, P_{m+2Bn+1}) \;\middle|\; 1 \leq i \leq n, 1 \leq j \leq 2B\right\}, \tag{3.20}$$

$$\mathcal{P}_{\rm neg} = \left\{(P_{k+2Bn}, P_{m+2Bn+1}) \;\middle|\; 1 \leq k \leq m\right\}. \tag{3.21}$$

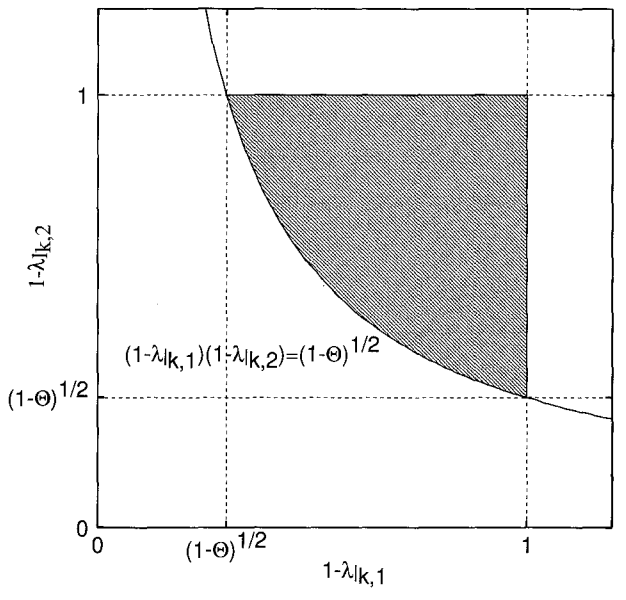

Figure 3.2: The region (the area colored by gray) where $1-\lambda_{l_{k,1}\alpha}$ and $1-\lambda_{l_{k,2}\alpha}$ exist when Inequality (3.25) is satisfied.

From $\mathcal{P}_{\text{pos}}$ and $\mathcal{P}_{\text{neg}}$, we have the following inequalities:

$$(1-\lambda_{x_i\alpha})(1-\lambda_{\bar{x}_i\alpha}) \leq 1-\Theta \quad \text{when } (P_{2B(i-1)+j}, P_{m+2Bn+1}) \in \mathcal{P}_{\text{pos}}, \tag{3.22}$$

$$(1-\lambda_{l_{k,1}\alpha})^2(1-\lambda_{l_{k,2}\alpha})^2 > 1-\Theta \quad \text{when } (P_{k+2Bn}, P_{m+2Bn+1}) \in \mathcal{P}_{\text{neg}}. \tag{3.23}$$

We can write these inequalities using Boolean variables θ_l as follows,

$$(3.22) \Rightarrow \theta_{x_i} \vee \theta_{\bar{x}_i}, \tag{3.24}$$

$$(3.23) \Leftrightarrow (1-\lambda_{l_{k,1}\alpha})(1-\lambda_{l_{k,2}\alpha}) > \sqrt{1-\Theta}. \tag{3.25}$$

Since both $\lambda_{l_{k,1}}$ and $\lambda_{l_{k,2}}$ are probabilities, the following formulas are always satisfied:

$$0 \leq 1-\lambda_{l_{k,1}} \leq 1, \quad 0 \leq 1-\lambda_{l_{k,2}} \leq 1. \tag{3.26}$$

As we see from Figure 3.2,

$$(3.25) \text{ and } (3.26) \Rightarrow 1-\lambda_{l_{k,1}} > \sqrt{1-\Theta} \wedge 1-\lambda_{l_{k,2}} > \sqrt{1-\Theta} \tag{3.27}$$

$$\Leftrightarrow \bar{\theta}_{l_{k,1}} \wedge \bar{\theta}_{l_{k,2}} \Leftrightarrow \overline{\theta_{l_{k,1}} \vee \theta_{l_{k,2}}}. \tag{3.28}$$

At first, I will show that the condition (a) of L-reduction is satisfied for this algorithm f. We compare $OPT(I)$ which is the maximum of clauses evaluated false of I, with $OPT(I')$, which is the maximum of satisfied inequalities of Inequalities (3.22) and (3.23) of I'.

Suppose that we have optimal values of $\lambda_{l\alpha}$ and Θ for $OPT(I')$. Then, we see that Inequalities (3.22) for all i are always satisfied. If the inequalities corresponding to a variable x_i of I are not satisfied, the optimal cost of $OPT(I')$ decreases by $2B$ because $2B$ inequalities of Inequalities (3.22) are generated by f from $2B$ different protein pairs. The cost increases by at most B because the variables $\lambda_{x_i\alpha}$ and $\lambda_{\bar{x}_i\alpha}$ concerning x_i of I appears at most B times in Inequalities (3.23). Note that the variable x_i also appears at most B times positively or negatively in clauses of I of MAX 2UNSAT-B. Since the total cost decreases by at least B, it is more profitable that Inequalities (3.22) for all i are always satisfied by optimal solutions of $OPT(I')$.

Accordingly, Boolean formulas 3.24 corresponding to variables x_i of I are always satisfied, either θ_{x_i} or $\theta_{\bar{x}_i}$ is always assigned to true, and both are never assigned to false simultaneously. In other words, these $\lambda_{x_i\alpha}$ and $\lambda_{\bar{x}_i\alpha}$ can not satisfy Inequalities (3.23) corresponding to clauses of I more than $OPT(I)$. Therefore,

$$OPT(I') \leq OPT(I) + 2Bn. \tag{3.29}$$

As we also see from Figure 3.2, even if $\bar{\theta}_{l_{k,1}} \wedge \bar{\theta}_{l_{k,2}}$ of Boolean formula 3.28 is true, the values of $\lambda_{l_{k,1}\alpha}$ and $\lambda_{l_{k,2}\alpha}$ do not always satisfy Inequality (3.23) corresponding to clauses C_k of I, and consequently, Inequality (3.29) may not be exactly equal.

We can consider the following assignments to variables $\lambda_{x_i\alpha}$ and $\lambda_{\bar{x}_i\alpha}$ of MAX PPI from the truth assignments of $OPT(I)$ of MAX 2UNSAT-B,

$$\begin{cases} x_i = \text{true} \Rightarrow \lambda_{x_i\alpha} = \Theta,\ \lambda_{\bar{x}_i\alpha} = 0 \\ x_i = \text{false} \Rightarrow \lambda_{x_i\alpha} = 0,\ \lambda_{\bar{x}_i\alpha} = \Theta. \end{cases} \tag{3.30}$$

These values satisfy exactly $OPT(I)$ inequalities of Inequalities (3.23) corresponding to clauses evaluated false by optimal solutions of I, and satisfy all of Inequalities (3.22). Therefore, $OPT(I')$ achieves the right-hand side of Inequality (3.29):

$$OPT(I') = OPT(I) + 2Bn. \tag{3.31}$$

From Corollary 1, $OPT(I)$ has at least a constant fraction of m such that $OPT(I) \geq m/\alpha_r$. Since the number of variables is less than twice the number of clauses, it holds that $n \leq 2m$ and $n \leq 2\alpha_r OPT(I)$. Substituting n from Equation (3.31) into this, we have

$$OPT(I') \leq (1 + 4B\alpha_r)OPT(I). \tag{3.32}$$

It follows that f satisfies the condition (a) in the definition of L-reduction with the constant $\alpha = 1 + 4B\alpha_r$.

Next, I show that the algorithm g which I will construct satisfies the condition (b) in L-reduction. Recall that, given the solution of I' with cost c', the function g has to produce the solution of I with cost c. In this case, given solutions of I and I', costs of I and I' correspond to the number of clauses satisfied and the number of inequalities that are satisfied, respectively. We can obtain truth assignments of θ_{x_i}s from the solution of I' in the previous manner. g assigns either true or false to each x_i on the basis of both assignments of θ_{x_i} and $\theta_{\bar{x}_i}$. And for each i in x_i and θ_{x_i}, we evaluate the difference of costs denoted by Δ_i, when we replace θ_{x_i} and $\theta_{\bar{x}_i}$ with x_i and $\bar{x}_i$, respectively. Note that $c - c' = \sum_{i=1}^{n} \Delta_i$.

(i) When $(\theta_{x_i}, \theta_{\bar{x}_i}) = (\text{true}, \text{true})$, g assigns true to x_i (and false to $\bar{x}_i$). Since $2B$ inequalities for i in Inequalities (3.22) are satisfied, the cost c decreases by $2B$. In Inequalities (3.23), c increases by at most B because $\bar{x}_i$ is assigned to false. In total, c decreases by at most $2B$, that is, $\Delta_i \geq -2B$.

(ii) When $(\theta_{x_i}, \theta_{\bar{x}_i}) = (\text{true}, \text{false})$ or $(\text{false}, \text{true})$, g assigns θ_{x_i} to x_i. If Inequalities (3.22) for i are satisfied, c decreases by $2B$. Otherwise, c is not changed. It follows that $\Delta_i \geq -2B$.

(iii) When $(\theta_{x_i}, \theta_{\bar{x}_i}) = (\text{false}, \text{false})$, g assigns true to x_i. Then, Inequalities (3.22) for i are not satisfied, and c decreases by at most the appearances of x_i. It follows that $\Delta_i \geq -B$.

Consequently, we have

$$c - c' = \sum_{i=1}^{n} \Delta_i \geq -2Bn \tag{3.33}$$

$$\Leftrightarrow \quad c \geq c' - 2Bn \tag{3.34}$$

$$\Leftrightarrow \quad c - OPT(I') \geq c' - 2Bn - OPT(I') \tag{3.35}$$

$$\Leftrightarrow \quad c - OPT(I) \geq c' - OPT(I') \text{ (from Equation (3.31))} \tag{3.36}$$

$$\Leftrightarrow \quad |c - OPT(I)| \leq |c' - OPT(I')| \text{ (because } c - OPT(I) \leq 0) \tag{3.37}$$

The condition (b) of the L-reduction is satisfied with $\beta = 1$. In consequence of the properties of f and g, MAX PPI is MAX SNP-hard. ∎

3.6 Time Complexity of LPBN Method

In the previous section, I showed that MAX PPI is MAX SNP-hard. In other words, it is intractable to maximize classification accuracy of protein-protein interactions. The result also says that there is no polynomial-time approximation algorithm within an arbitrary ratio. Therefore, heuristic algorithms such as the LPBN method based on linear programming are necessary for inferring protein-protein interactions. Since we can not design classification accuracy as an objective function of linear programming problems, I used the summation of errors for every protein pair in the LPBN method instead. We have not obtained the approximate guarantee for the LPBN method. In general, it is known that we can obtain, if any, optimum solutions of linear programming problems in polynomial time using solvers such as interior-point methods [43] [36].

3.7 Comparison with Induction of Oblique Decision Trees

MAX PPI can be similar to a known NP-complete problem called the induction of oblique decision trees (IODT) [21, 34] when we take another look at MAX PPI from a different point of view. By taking logarithms of both sides of inequalities in MAX PPI, we have the following linear inequalities,

$$\begin{aligned} \sum_{D_{mn} \in P_{ij}} \gamma_{mn} &\leq \beta \quad \text{if } (P_i, P_j) \in \mathcal{P}_{\text{pos}}, \\ \sum_{D_{mn} \in P_{ij}} \gamma_{mn} &> \beta \quad \text{if } (P_i, P_j) \in \mathcal{P}_{\text{neg}}, \end{aligned} \tag{3.38}$$

where $\gamma_{mn} = \ln(1 - \lambda_{mn})$ and $\beta = \ln(1 - \Theta)$. Let M' denote the number of all domain pairs, and M' is set to $M(M+1)/2$. For each protein pair (P_i, P_j), I construct a vector $\mathbf{v}_{ij} (\in R^{M'})$ such that each element of $\mathbf{v}_{ij}$ corresponding to a domain pair (D_m, D_n) is defined as

$$\mathrm{v}_{ij}^{(mn)} = \begin{cases} 1 & \text{if } D_{mn} \in P_{ij}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.39}$$

In this setting, MAX PPI is equivalent to a problem to find a hyperplane (γ, β) which splits examples (vectors) into positive and negative ones such that the number of misclassified examples with γ is minimized. The optimal hyperplane (γ, β) is described as

$$\gamma \cdot \mathbf{v} = \beta, \tag{3.40}$$

where γ $(\in R^{M'})$ is a vector with γ_{mn} and $\mathbf{v} \in R^{M'}$.

On the other hand, IODT [21, 34] is defined in a similar manner, but the objective function e (called the sum-minority measure) to be minimized is different. As we have seen, a hyperplane divides a set of examples into two subsets, which we call X_1 and X_2, respectively. For brevity, let the number of examples in $\mathcal{P}_{\mathrm{pos}}$ $(\mathcal{P}_{\mathrm{neg}})$ in X_1 be u_1 (v_1), and the number of examples in $\mathcal{P}_{\mathrm{pos}}$ $(\mathcal{P}_{\mathrm{neg}})$ in X_2 be u_2 (v_2). Then, IODT is defined as the following: given a positive point set $\mathcal{P}_{\mathrm{pos}}$, a negative point set $\mathcal{P}_{\mathrm{neg}}$ and a value k, find a hyperplane (γ, β) in Equation (3.40) such that $e \leq k$, where $e = \min(u_1, v_1) + \min(u_2, v_2)$. It is known that the problem of determining if there is a hyperplane (γ, β) that satisfies $e \leq k$ is NP-complete [21].

There are two major differences between MAX PPI and IODT. One is the constraints on parameters in MAX PPI. That is, coefficients $\gamma_{mn} = \ln(1 - \lambda_{mn}) \leq 0$ and $\beta = \ln(1 - \Theta) \leq 0$ of the hyperplane in MAX PPI can take only non-positive values. By these constraints, the intractabilities of MAX PPI may differ from that of IODT. Recall that MAX PPI is MAX SNP-hard as well as NP-hard, and IODT is NP-complete.

The other difference is the scores of the objective functions. The objective function of IODT may result in assigning the same label to all examples. We can suppose that $\gamma \cdot \mathbf{v}_{ij} \leq \beta$ holds for any example $\mathbf{v}_{ij}$ in X_1 without loss of generality. For simplicity, we consider here two dimensional space and

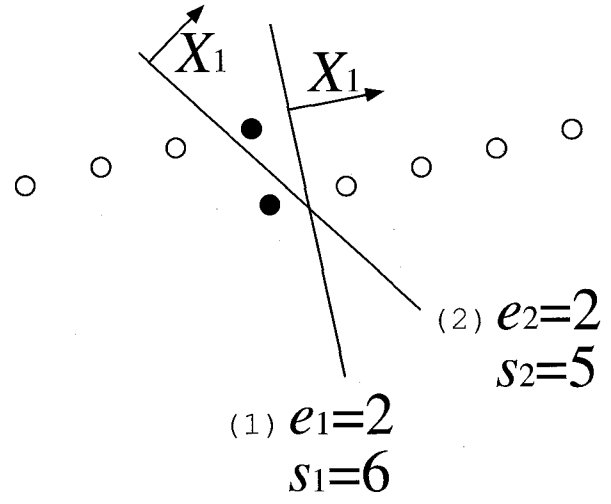

Figure 3.3: Possible hyperplanes (lines) that split examples. The open circles belong to class $\mathcal{P}_{\text{pos}}$ and the filled ones belong to class $\mathcal{P}_{\text{neg}}$.

only one hyperplane (line) that splits examples for IODT. In Figure 3.3, each of two lines (1) and (2) splits seven positive and two negative examples. Let e_i (s_i) be the score of the objective function of IODT (MAX PPI) with line (i). Recall that IODT uses the sum-minority measure $e = \min(u_1, v_1) + \min(u_2, v_2)$ and MAX PPI uses the sum $s = v_1 + u_2$ as the objective functions. We then obtain $e_1 = \min\{4, 0\} + \min\{3, 2\} = 2$, $e_2 = \min\{4, 1\} + \min\{3, 1\} = 2$, $s_1 = 4 + 2$, and $s_2 = 4 + 1$. In this example, IODT can choose one of the two lines. Moreover, the score of the sum-minority measure is always 2 with any line in Figure 3.3, and IODT assigns positive labels to all examples with some of the lines like lines (1) and (2). In other words, these lines do not contribute to the classification. However, MAX PPI always chooses line (1) with the maximum score $s_1 = 6$ among the two lines.

Chapter 4

Application toward Inference of the Strengths of Protein-Protein Interactions

In previous chapters, I developed several methods for inferring protein-protein interactions.

Recently, large-scale two-hybrid systems were developed for comprehensive analysis of protein-protein interactions in *Saccharomyces cerevisiae* [23, 24, 38]. However, there was a large gap between the results by Ito et al. [23, 24] and Uetz et al. [38].

In Ito's experiment, multiple experiments were performed for the same protein pairs in practice and thus the ratio of the number of observed interactions to the number of experiments is available for each protein pair. Therefore, it is reasonable to use the ratio as input data. We regard it as the *strength* of interactions in this thesis. In this chapter, I propose a new method, called LPNM (Linear Programming for NuMerical interaction data), for inferring strengths of protein-protein interactions by applying the technique of the LPBN method. I show that it outperforms other existing methods. In addition, I propose a faster method than the LPNM method, called ASNM (ASsociation method for NuMerical interaction data), and verify their elapsed times.

4.1 Algorithms

4.1.1 LPNM: LP-based Method for Numerical Interaction Data

Here I describe an LP-based method for numerical interaction data.

In the LPBN method, we used some threshold Θ to predict protein-protein interactions. On the other hand, in the LPNM method, we set ρ_{ij} to be the ratio of interactions between proteins P_i and P_j in a series of experiments, that is,

$$\rho_{ij} = \frac{N_{ij}}{Z}, \tag{4.1}$$

where N_{ij} is the number of times an interaction between proteins P_i and P_j is observed in the experiments, and Z is the total number of experiments.

Since ρ_{ij} is the ratio of interactions between P_i and P_j, we consider here to minimize the difference between $\Pr(P_{ij} = 1)$ and ρ_{ij}, in other words, the difference between the probability of observing an interaction in the above probabilistic model and the ratio of the interactions observed in the experiments.

When $\Pr(P_{ij} = 1)$ and ρ_{ij} are equivalent, the following holds:

$$\sum_{D_{mn} \in P_{ij}} \ln(1 - \lambda_{mn}) \;=\; \ln(1 - \rho_{ij}). \tag{4.2}$$

From the above equation, we have a linear equation

$$\sum_{D_{mn} \in P_{ij}} \gamma_{mn} = \beta_{ij} \tag{4.3}$$

for any $P_{i,j}$ by setting

$$\gamma_{mn} \;=\; \ln(1 - \lambda_{mn}), \tag{4.4}$$

$$\beta_{ij} \;=\; \ln(1 - \rho_{ij}). \tag{4.5}$$

If we have γ_{mn} for any m and n satisfying the above equations, we can obtain parameters for domain-domain interactions consistent with a numerical interaction data set.

These equations, however, do not always hold. It is hence reasonable to try to minimize the sum of the difference

$$\sum_{o_{ij} \in \mathcal{O}} \left| \sum_{D_{mn} \in P_{ij}} \gamma_{mn} - \beta_{ij} \right|. \tag{4.6}$$

We therefore use the following linear program to minimize the difference:

$$\begin{aligned}
\text{minimize} \quad & \sum_{o_{ij} \in \mathcal{O}} \alpha_{ij}\,, \\
\text{subject to} \quad & \sum_{D_{mn} \in P_{ij}} \gamma_{mn} - \beta_{ij} \leq \alpha_{ij}, \\
& \beta_{ij} - \sum_{D_{mn} \in P_{ij}} \gamma_{mn} \leq \alpha_{ij}, \\
& \gamma_{mn} \leq 0 \quad \text{for all } \gamma_{mn}, \\
& \alpha_{ij} \geq 0 \quad \text{for all } \alpha_{ij}, \\
& \beta_{ij} < 0.
\end{aligned}$$

4.1.2 ASNM: Association Method for Numerical Interaction Data

I propose a new simple method for inferring the strength of protein-protein interactions, which we call the ASNM method. This method is derived by extending the association method [37] (for binary interaction data) into one for numerical interaction data. The association method uses the number of interacting protein pairs (I_{mn}) to infer the score (probability of interaction) for (D_m, D_n). In the ASNM method, we use the summation of the strengths (ρ_{ij}) of interaction between P_i and P_j instead of I_{mn}, where the protein pair (P_i, P_j) includes the target domain pair (D_m, D_n). I then define the score $ASNM(D_m, D_n)$ for (D_m, D_n) as

$$ASNM(D_m, D_n) \;=\; \frac{1}{N_{mn}} \sum_{\{o_{ij} \in \mathcal{O} | D_{mn} \in P_{ij}\}} \rho_{ij}. \tag{4.7}$$

Recall that N_{mn} is the number of protein pairs containing domain pairs (D_m, D_n). If the ratio ρ_{ij} for each protein pair (P_i, P_j) always takes either 0 or 1, $ASNM(D_m, D_n)$ becomes equivalent to the score $ASSOC(D_m, D_n)$ in the association method because it holds that $I_{mn} = \sum_{\{o_{ij} \in \mathcal{O} | D_{mn} \in P_{ij}\}} \rho_{ij}$.

Although the LPNM method minimizes the summation of errors, it seems to be considered that the ASNM method becomes a maximum likelihood estimate on a probability distribution.

4.2 Data and Implementation

I compared the LPNM and ASNM method with the association method (ASSOC) and the EM method (EM). For the training and test data of protein-protein interactions, I used the full data of Ito's Yeast Interacting Proteins (YIP) database [23, 24]. The main reason for using this database is that the YIP database also provides numerical interaction data for pairs of proteins as the number of IST (Interaction Sequence Tag) hits. For each protein in this database, I obtained its sequence data from the Swissprot/TrEMBL database [4] in the same way as was done with the DIP database in chapter 2. In order to derive domains from the sequences, I used InterProScan (version 3.1) [45] again.

I used glpsol (version 4.4) on Linux (http://www.gnu.org/software/glpk/) for solving linear programs. The experiments were mostly performed on a PC cluster with 8 Pentium Xeon 2.8 GHz processors, where only one was used in all experiments.

As in chapter 2, the scores obtained by ASSOC were used as the initial values of λ_{mn} for EM, and EM steps were repeated until the difference of log-likelihood between two consecutive steps became less than 0.01 or until the number of repeats exceeded 200, with values of $fp = 2.5 \times 10^{-4}$ and $fn = 0.80$ used for EM.

I evaluated the methods by root mean squared error (RMSE) between the predicted probability $\Pr(P_{ij} = 1)$ and the observed ratio ρ_{ij} from the YIP database. To be precise, for a set of protein pairs $\mathcal{P}$,

$$RMSE = \sqrt{\frac{1}{|\mathcal{P}|} \sum_{P_{ij} \in \mathcal{P}} (\Pr(P_{ij} = 1) - \rho_{ij})^2}.$$

4.3 Results

Here, I show results on numerical interaction data. I evaluated LPNM, ASNM, EM and ASSOC by 5-fold cross validation. I used 1,586 interaction pairs of proteins and the numbers of their IST hits as a whole data set.

In numerical interaction data, the ratio of the number of IST hits to the number of experiments is given for each pair of proteins. On the other hand, EM and ASSOC require labels (positive (interact) or negative (not interact)) to find appropriate parameters. We then must set some threshold to divide the set of protein pairs into positive and negative data. I set here the threshold for IST hits to be 3, that is, interaction pairs whose IST hits are less than 3 are regarded as negative data, and the others (those pairs with ≥ 3 hits) as positive data. This threshold might seem to be too small compared with the total number of experiments ($192 = 96 \times 2$). However, the numbers of IST hits for most protein pairs are very low and thus I used this threshold. I examined several other threshold values, but the results did not change significantly.

Table 4.1 shows root mean squared errors and average elapsed time for test data sets using LPNM, ASNM, EM and ASSOC. It should be noted that I employed 5-fold cross validation and the k-th row means that the k-th block among the five blocks of the data was used as a test data set.

It is seen from the table that the errors of both LPNM and ASNM for test data sets are quite similar, and much smaller than those of ASSOC and EM. Since the strength (i.e., the ratio of the number of IST hits to the number of experiments) takes a value between 0.0 and 1.0, the errors of LPNM and ASNM are considerably small whereas the errors for EM and ASSOC are large. These results suggest that, in the sense of minimizing RMSE, LPNM and ASNM was able to find much better parameters (i.e., probabilities of domain-domain interactions) than existing methods. It is reasonable because LPNM and ASNM try to minimize the error, whereas EM or ASSOC do not try to minimize the error.

The average errors of ASNM for test data sets are slightly worse than that of LPNM, and the error of LPNM for training data sets are considerably

Table 4.1: Root mean squared errors and average training elapsed time of LPNM, ASNM, EM and ASSOC for numerical interaction data.

		LPNM		ASNM	
		Train	Test	Train	Test
Error	1st	0.0103880	0.0312939	0.0365687	0.0408624
	2nd	0.0145225	0.0329882	0.0381153	0.0480632
	3rd	0.0143729	0.0347589	0.0429533	0.0471907
	4th	0.0141168	0.0282775	0.0397846	0.0356935
	5th	0.0140418	0.0266282	0.0424590	0.0306575
	Average	0.0134884	0.0307893	0.0399762	0.0404935
Time (sec)		1.203068	-	0.0077122	-

		EM		ASSOC	
		Train	Test	Train	Test
Error	1st	0.470687	0.327673	0.452380	0.315208
	2nd	0.479989	0.339117	0.455613	0.308925
	3rd	0.484887	0.315147	0.455444	0.290413
	4th	0.476884	0.251272	0.453617	0.241639
	5th	0.495042	0.242480	0.467038	0.227669
	Average	0.481498	0.295138	0.456818	0.276771
Time (sec)		1.620078	-	0.0088252	-

smaller than that of ASNM. This suggests that LPNM may overfit in this case.

It is also seen that the error for EM is always greater than that for ASSOC. This is reasonable because EM tries to make the probabilities for interacting pairs in the training data close to 1.0 whereas strengths of most interacting pairs are much lower than 1.0.

Errors for training data sets are smaller than those for test data sets generally. However, the errors of ASSOC for training data sets are larger than those for test data sets. This can be considered because although the errors are calculated for their strengths, ASSOC uses just binary data (whether or

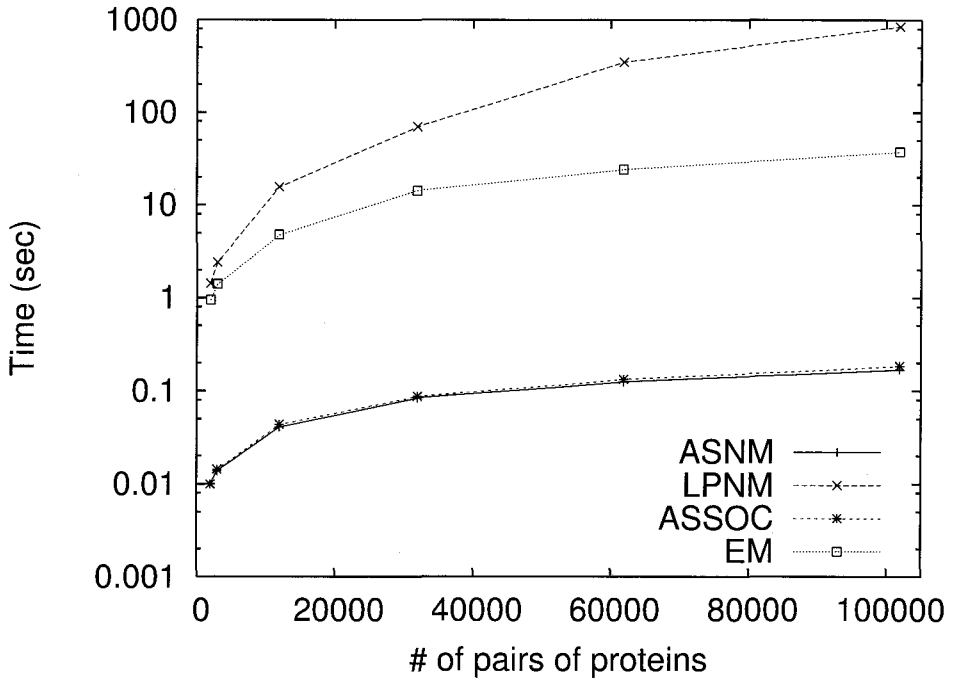

Figure 4.1: Elapsed time (log scale) for training in ASNM, LPNM, EM and the association method. The X-axis shows the number of input data sets, which is the number of protein pairs. The Y-axis shows the logarithm of elapsed time.

not each pair of proteins interacts), and does not use those strengths.

Figure 4.1 shows elapsed time of training for ASNM, LPNM, EM and ASSOC. It is seen from the figure that the elapsed times of ASNM and ASSOC are much smaller than that of LPNM and EM, and that the time of LPNM increases more steeply than those of ASNM and ASSOC when the number of input data sets increases.

To see the distributions of errors for the methods, I plotted the number of proteins according to the error between the ratio in the data set and predicted one in Figure 4.2. It shows the average frequencies of probability errors of protein-protein interactions for the test data during the cross validation by LPNM, ASNM, EM and ASSOC, respectively. Note that distributions of errors for EM (and ASSOC) are large around 1.0 whereas these are small for

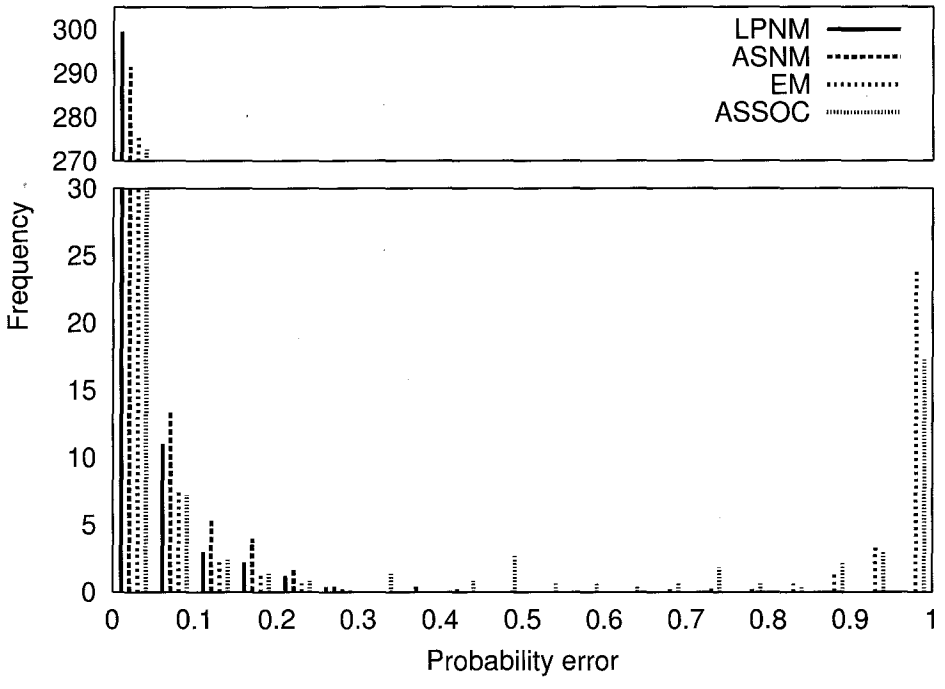

Figure 4.2: Distributions of probability errors for LPNM, ASNM, EM and ASSOC. The Y-axis shows the number of interacting protein pairs for which the errors (between the predicted probabilities and the observed probabilities) are within the specified range. The average numbers over 5 test data sets are shown. I omit the range of frequencies between 30 and 270.

Table 4.2: Examples of inferred number of IST hits by LPNM, EM and ASSOC.

Protein pair		YIP	LPNM	EM	ASSOC
Q06178	P53204	36	19	192	192
Q12518	Q99210	23	14	192	192
P53949	P50946	23	5	192	192
P32458	P32468	11	1	0	0
P27472	P47011	11	11	192	192
P07278	P05986	10	4	192	192
Q04739	P12904	9	3	192	192
P40054	P40054	9	3	191	187
P40917	P32366	7	15	192	192
P36017	P50079	7	2	0	0
P25383	Q99303	7	1	192	87
P23291	P39010	7	5	192	192
Q12084	Q12084	6	0	192	192
Q06169	Q12402	6	6	192	192
Q02821	P40892	6	1	0	0
P38697	Q02821	6	2	186	144

LPNM and ASNM. This is reasonable because EM and ASSOC use either 0 or 1 as probabilities of interactions instead of strengths ρ_{ij} in LPNM and ASNM. Additionally, EM tries to maximize the probabilities for interacting protein pairs, but the real probabilities are small.

Table 4.2 shows examples of inferred strengths (the number of IST hits) of protein-protein interactions for LPNM, EM and ASSOC. In this table, data are shown for protein pairs (in one test data set) for which the number of IST hits in the YIP database are greater than 5 and at least one method output non-zero probabilities. It can be seen that inferred numbers of IST hits by LPNM are much closer to the numbers in the YIP database than those inferred by EM and ASSOC. It is also seen that in most cases, the inferred numbers by EM and ASSOC are close to the maximum number of

Table 4.3: Overlapping rates of the core data by Ito et al. or interactions reestimated by LPNM against two data sets, interaction data by Uetz et al. and DIP core data (ScereCR20040404.tab), respectively.

	Uetz et al. (%)	DIP core (%)
Ito et al.	13.9	25.3
LPNM	14.7	32.0

IST hits (i.e., $192 = 96 \times 2$).

Table 4.3 shows overlapping rates of the core data by Ito et al. [23, 24] or interactions reestimated by LPNM against two data sets: the interaction data by Uetz et al. [38] and DIP core data (ScereCR20040404.tab (See Section 2.4)), respectively. I used all IST data from Ito, and estimated the probabilities of domain-domain interactions using the LPNM method. I predicted IST hits for all protein pairs of Ito's data.

We see from this table that the overlapping rate of Ito's original data against Uetz's data slightly increased due to the modification by LPNM. Moreover, the rate against DIP core data largely increased by LPNM. This overlap suggests that the LPNM method is an effective method to improve biological experimental results using yeast two-hybrid systems.

4.4 Discussion

I proposed an LP-based method (LPNM) and a simple method (ASNM) for inferring strengths of protein-protein interactions from experimental data. I compared the proposed methods with existing methods such as the association method and the EM method. For numerical interaction data, the LPNM and ASNM method outperformed existing methods. The ASNM method ran much faster than the LPNM method, and achieved almost the same accuracy as that obtained by the LPNM method.

The most important feature of the proposed methods is that strengths of protein-protein interactions are taken into account for both training and test data. Although most of the existing methods output scores ($\approx$ strengths)

of protein-protein interactions, training data was given as binary data. It seems difficult to modify the EM method so that numerical interaction data can be given as training data.

The LPNM method also has the feature similar to the LPBN method that several kinds of constraints can be put on. In this chapter, we used constraints on the strengths of interactions.

Though the LPNM method outperformed existing methods for numerical interaction data, its performance is not satisfactory as seen from Table 4.2. Therefore, improved methods for numerical data should be developed.

Chapter 5

A Model of Protein Evolution Using Domain Information

In previous chapters, we considered individual pairs of proteins based on domain information. In this chapter, we will focus on whole proteins based on domains from an evolutional point of view. First, I will define a network graph of protein domains, and some network measures. Next, I will show some figures of protein domain networks for some species, and that these networks have two types of power laws.

5.1 Protein Domain Network

I define a network graph $G(V, E)$ of protein domains. $\mathcal{P}$ is a set of proteins $\{P_i\}$, and $\mathcal{D}$ is a set of domains $\{D_m\}$. Each protein P_i has some domains D_m. A vertex v_i of V means a protein P_i. An edge e_n of E between two vertices v_i and v_j is connected if and only if both of the proteins P_i and P_j have a common domain D_m.

5.1.1 Network Measures

We define some measures for comparing and characterizing different networks. I plot their distributions for protein domain networks.

(1) Degree (k): The number of edges to which a vertex connects. It is also

called connectivity.

(2) Edge weight (w): In protein domain networks, the number of domains contained in both of two proteins.

(3) Vertex strength

(3a) Total weight (s): The summation of edge weights.

(3b) Strength (d): In protein domain networks, the number of domains contained in a protein.

5.1.2 Experimental Data

I measured some protein domain networks for some species. For information about proteins and domains, I used UniProt knowledgebase [2] (version 2.5). It provides a stable, comprehensive, non-redundant sequence collection, and the protein sequences are fully classified, richly and accurately annotated. For each sequence entry of UniProt, the following annotations are added: the sequence data, the citation information, the taxonomic data, functions of the protein, posttranslational modifications, domains, sites, secondary structure, quaternary structure, similarities to other proteins, diseases associated with any number of deficiencies in the protein, and sequence conflicts. Among these annotations, I used taxonomic data to analyze protein domain networks for specific species. Here, I chose six major species: Homo sapiens, Mus musculus, Drosophila melanogaster, Saccharomyces cerevisiae, Escherichia coli and Arabidopsis thaliana. For domain information, I used annotations of references to the protein domain databases InterPro[45], Pfam[7], SMART[29], ProDom[11], PROSITE[22] and PRINTS[3].

InterPro is an integrated database, and combines a number of databases which use two different main methods. One is sequence-motif methods. The PROSITE database is based on regular expressions and profiles of domains. Databases such as Pfam and SMART keep hidden Markov models. PRINTS provides fingerprints and groups of aligned, un-weighted motifs. Also provided are sequence-cluster methods. ProDom uses PSI-BLAST to cluster homologous domains. It is relatively comprehensive because they do not depend on manual crafting and validation of family discriminators.

I will show figures of frequency distributions using these domain databases. However, I will use only InterPro as a representative of those databases after comparing them because it is complicated to use all databases.

Note that the UniProt knowledgebase consists of two sections. One section is Swiss-Prot, which contains manually-annotated records with information extracted from literature and curator-evaluated computational analysis. Another section is TrEMBL, which has computationally analyzed records that await full manual annotation. I used only proteins of the Swiss-Prot section because the TrEMBL section may include uncertain annotations.

5.1.3 Results

I plotted some frequency distributions ($f(k)$, $f(w)$ and $f(s)$) of degree k, edge weight w, and total weight s for protein domain networks using proteins of the Swiss-Prot section in UniProt, and the annotations of references to some domain databases.

First, I plotted $f(k)$ of H. sapiens with domain databases InterPro, Pfam, SMART, ProDom, PROSITE, and PRINTS in order to compare them (see Figures 5.1, 5.2, 5.3, 5.4, 5.5 and 5.6). In the figures, each x-axis (log scale) means degrees k of vertices, and each y-axis (log scale) means the frequency $f(k)$ of vertices with degree k. We see that the shapes of the plots are very similar although there are differences of the number of data between those databases. Therefore, I use InterPro as their representative.

Figures 5.7, 5.8, 5.9, 5.10, 5.11 and 5.12 show frequency distributions $f(k)$ for six species: H. sapiens, M. musculus, D. melanogaster, S. cerevisiae, E. coli and A. thaliana, respectively. Table 5.1 shows the number of proteins and domains of each species in InterPro, Pfam, and SMART.

Several biological networks are known as scale-free. For example, networks of protein-protein interactions for Saccharomyces cerevisiae[25] and metabolic networks[41] have been published. Scale-free networks have scaling properties that the probability $P(k)$ that a vertex in the network is connected to k other vertices decays as the following power law:

$$P(k) \propto k^{-\gamma}, \tag{5.1}$$

where γ is a constant.

Table 5.1: The number of proteins and domains in InterPro, Pfam and SMART.

Species	The number of domains			The number of proteins
	InterPro	Pfam	SMART	
H. sapiens	3936	2354	497	11387
M. musculus	3545	2126	470	8166
D. melanogaster	1571	1037	259	2087
S. cerevisiae	2412	1631	282	4980
E. coli	2162	1352	93	3335
A. thaliana	1467	945	116	2925

We see that all the figures of frequency distributions of degree k show two types of power-law tendencies. One is a power law for low degrees of vertices with negative exponents, $-\gamma \simeq -1$. Another is a power law for high degrees of vertices with positive exponents, $-\gamma \simeq 1$. I will propose a model to reconstruct these properties in next sections.

I also plotted frequency distributions for other network measures including the edge weight w, (Figures 5.13, 5.14, 5.15, 5.16, 5.17 and 5.18) and two kinds of strengths s (Figures 5.19, 5.20, 5.21, 5.22, 5.23 and 5.24) and d (Figures 5.25, 5.26, 5.27, 5.28, 5.29 and 5.30), where s means the summation of weights of edges for a vertex, and d means the number of domains for a vertex, or a protein.

The distributions of the edge weights w and the strengths d as well as protein-protein interaction networks show single power-law behaviors with negative exponents. On the other hand, the distributions of the other strength s shows almost the same power-law behaviors as those of degrees k. It is reasonable that the summation of weights of edges for a vertex is almost same as the number of edges for the vertex because almost all weights of edges are one as we see from the figures of the edge weights w.

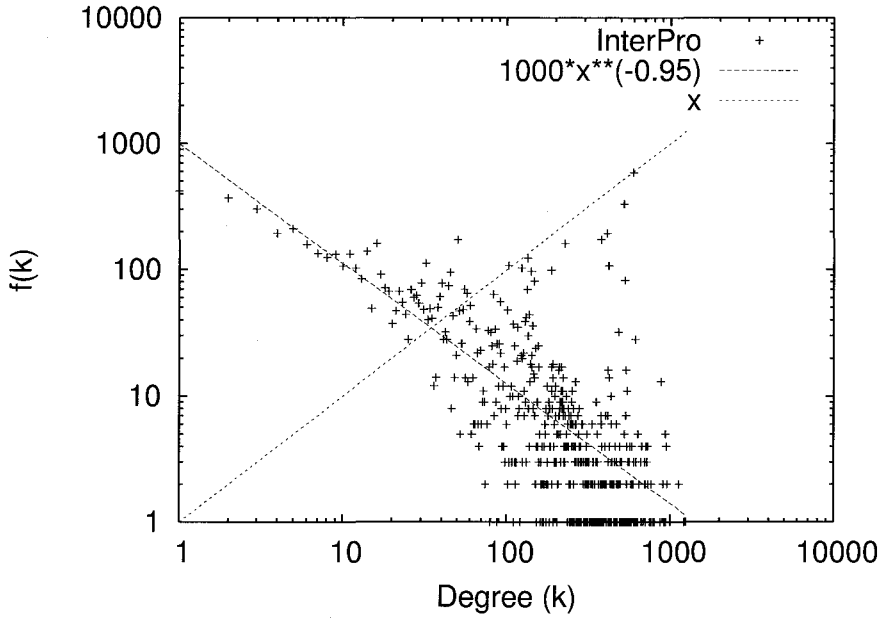

Figure 5.1: Homo sapiens (k) using InterPro

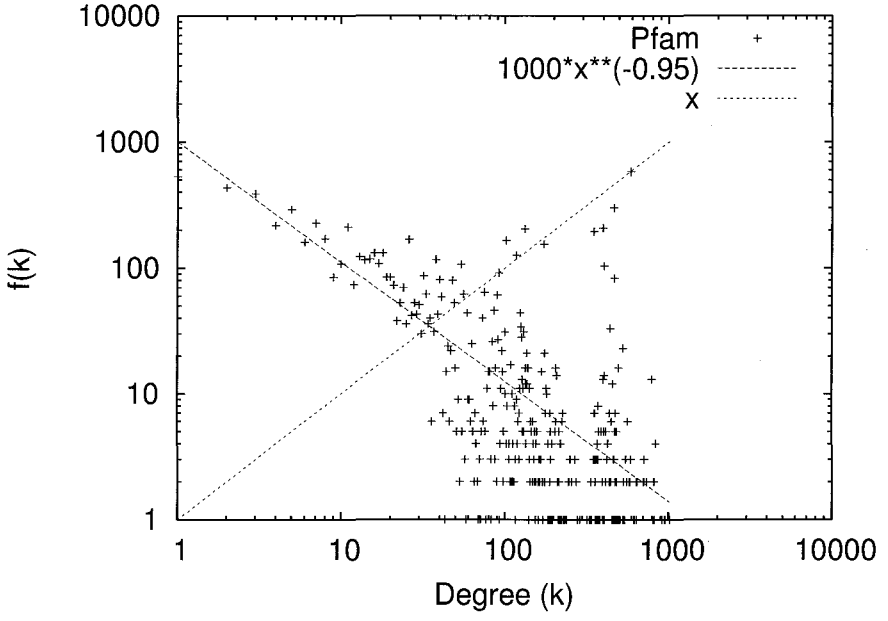

Figure 5.2: Homo sapiens (k) using Pfam

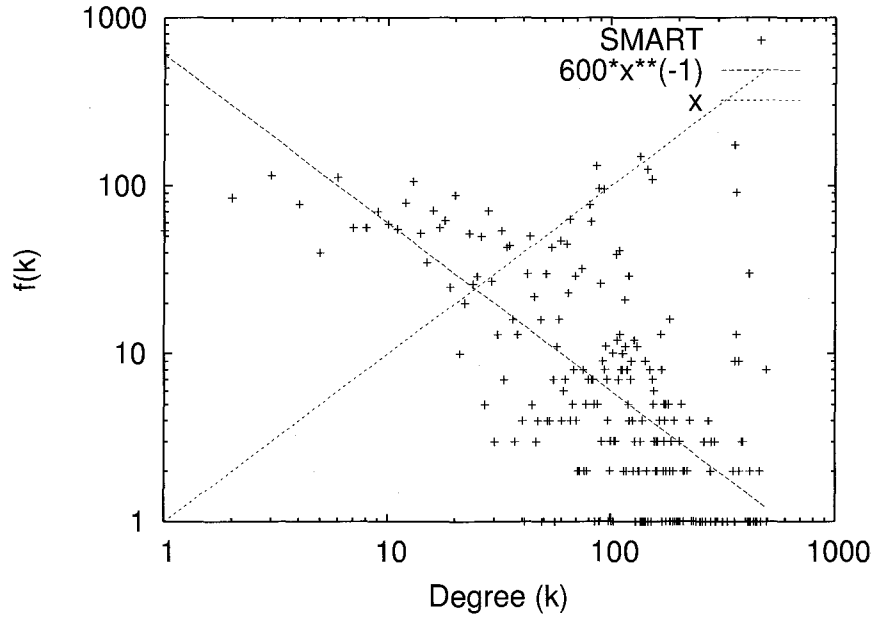

Figure 5.3: Homo sapiens (k) using SMART

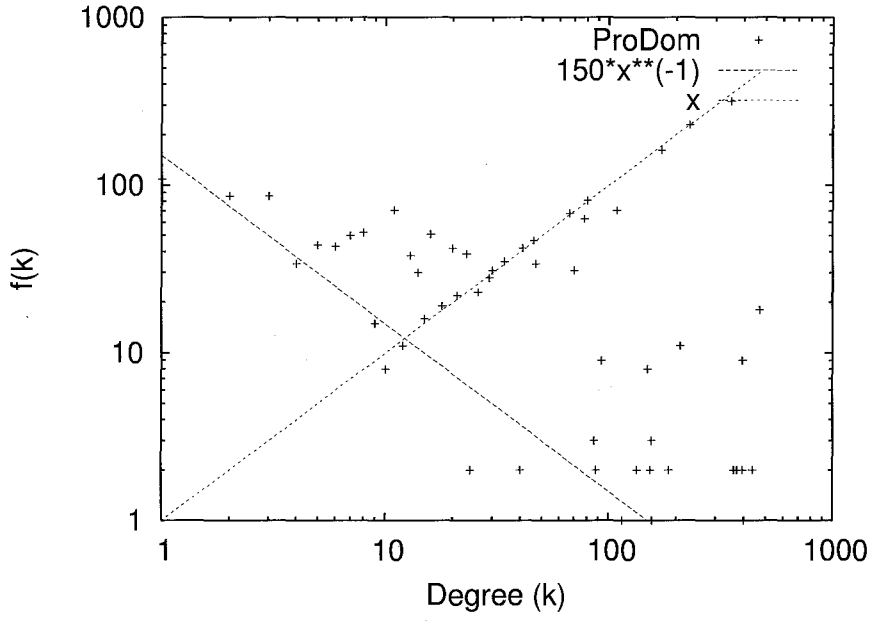

Figure 5.4: Homo sapiens (k) using ProDom

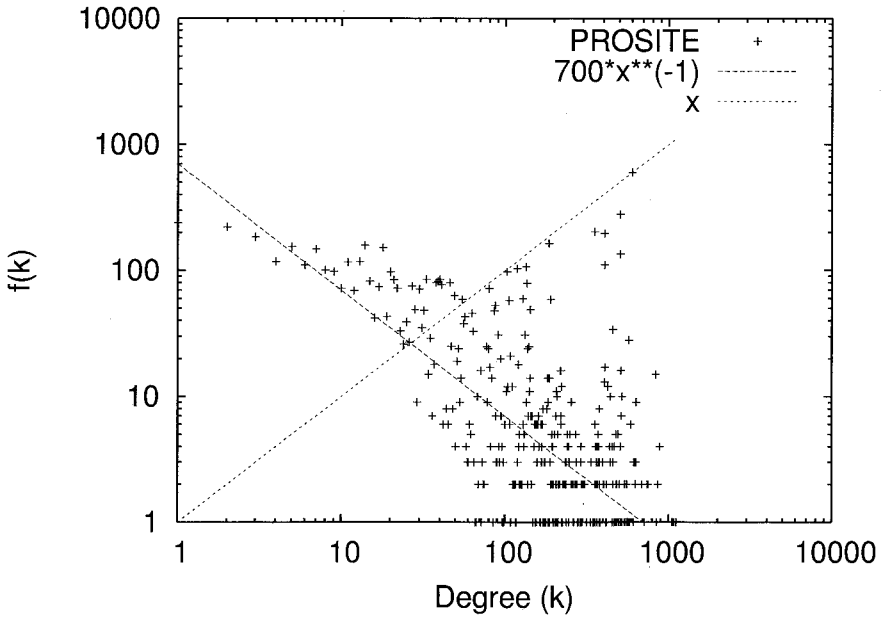

Figure 5.5: Homo sapiens (k) using PROSITE

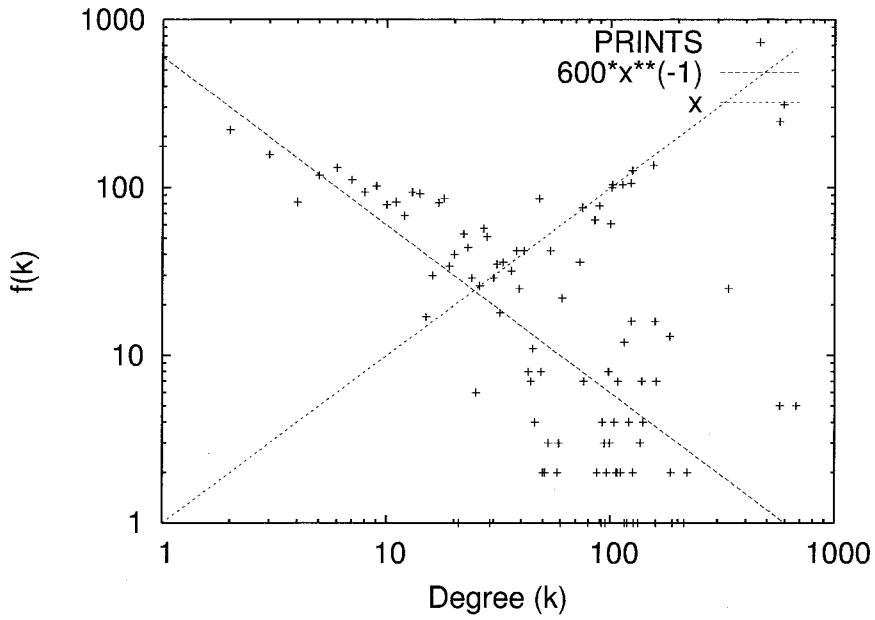

Figure 5.6: Homo sapiens (k) using PRINTS

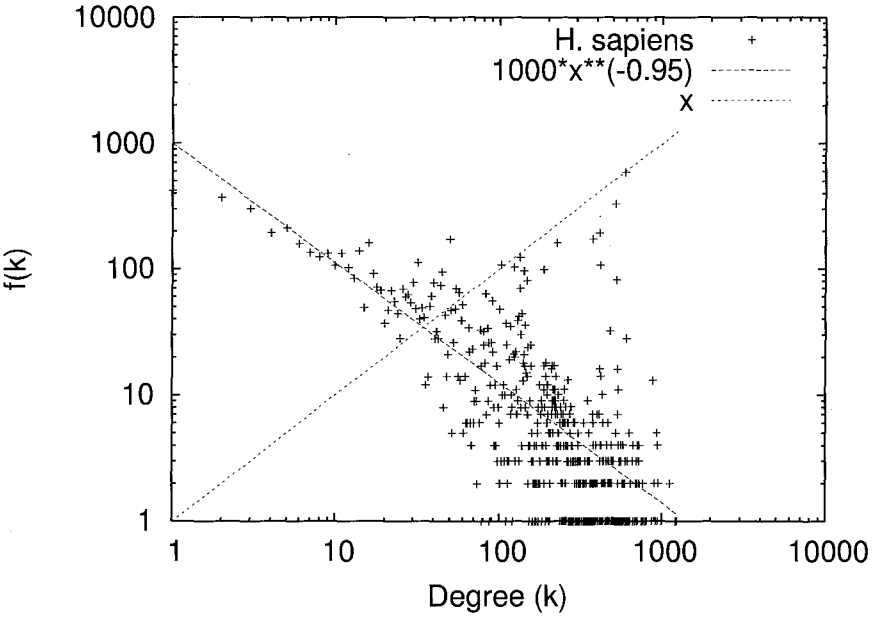

Figure 5.7: Homo sapiens (k)

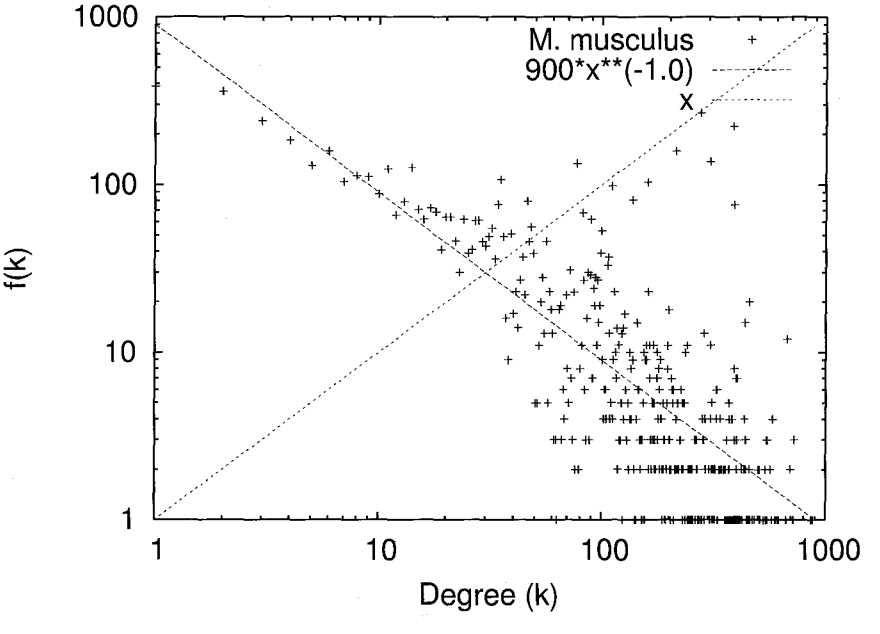

Figure 5.8: Mus musculus (k)

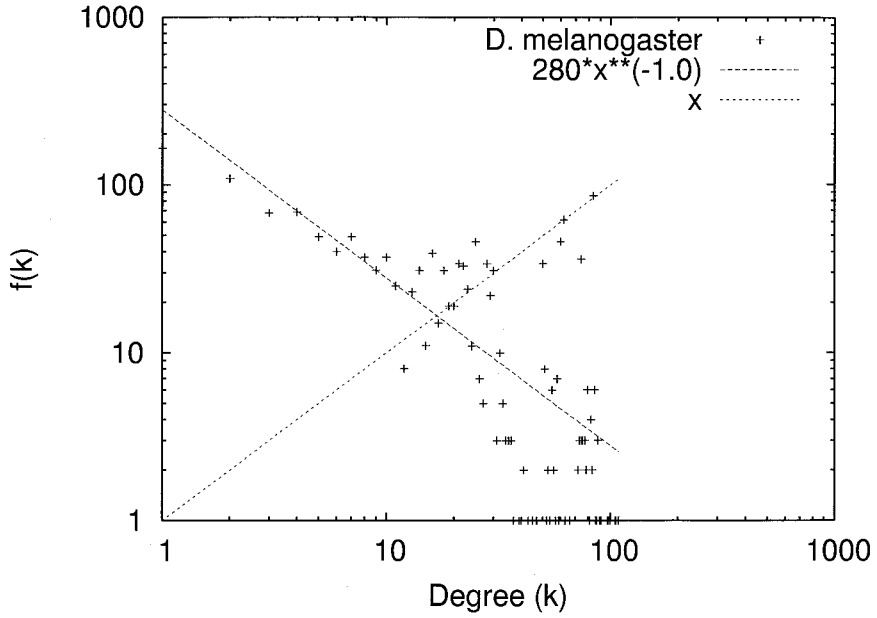

Figure 5.9: Drosophila melanogaster (k)

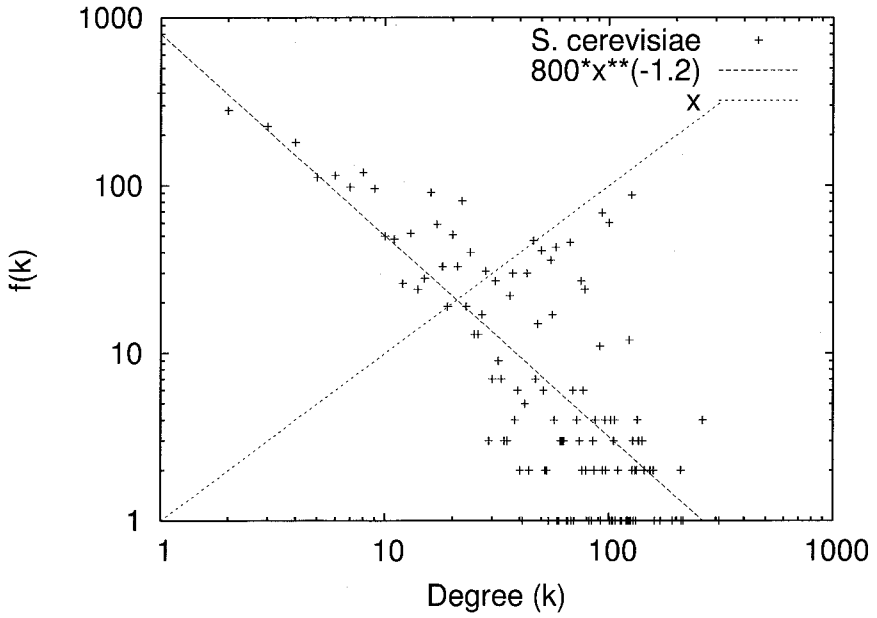

Figure 5.10: Saccharomyces cerevisiae (k)

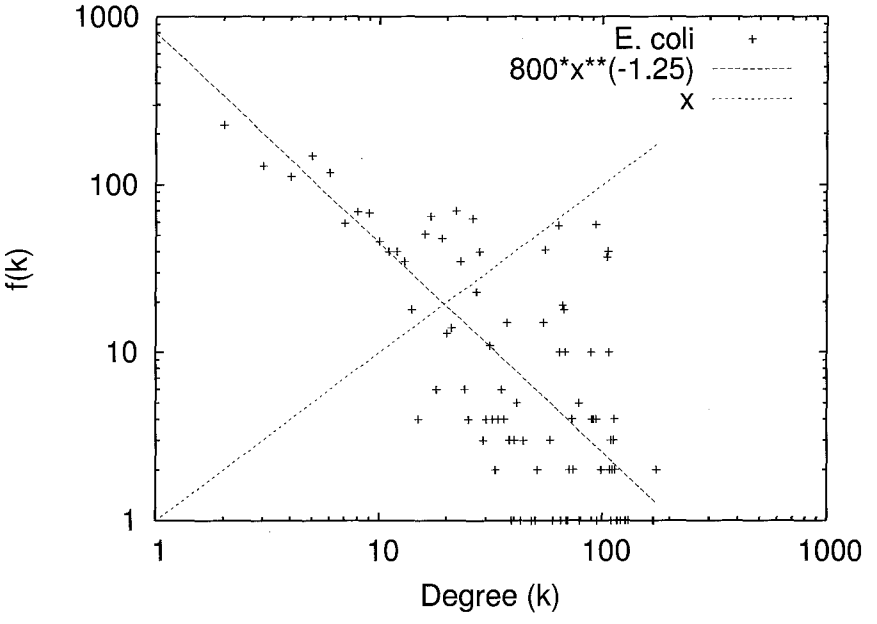

Figure 5.11: Escherichia coli (k)

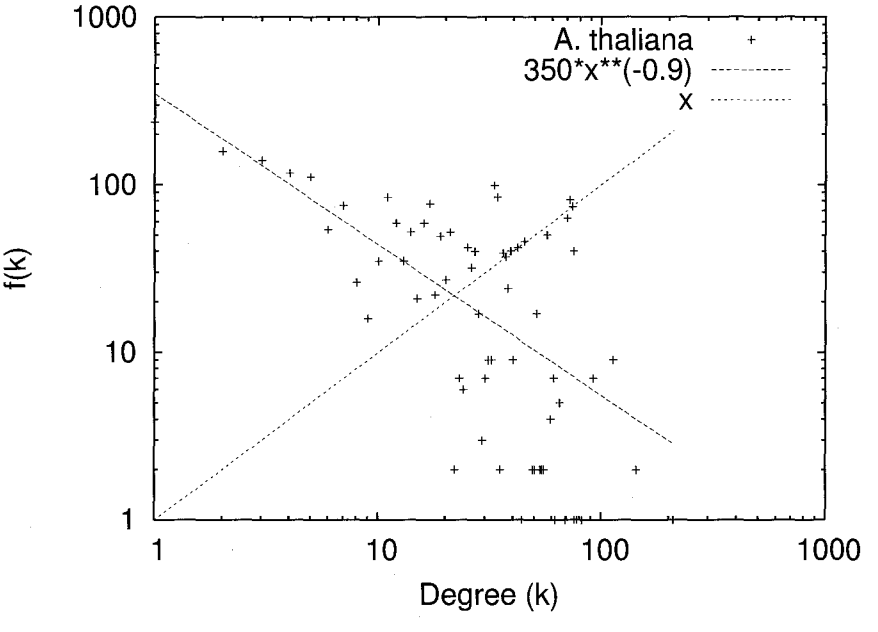

Figure 5.12: Arabidopsis thaliana (k)

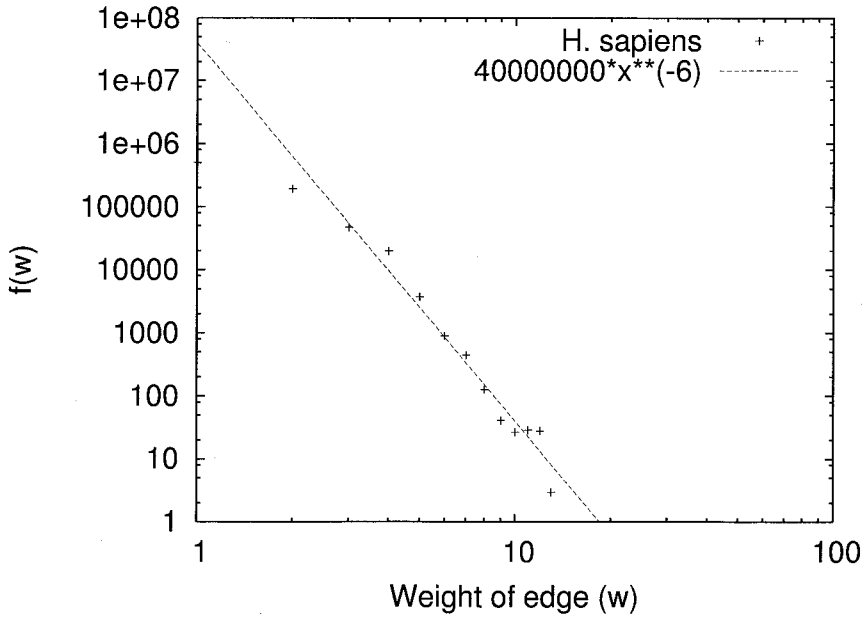

Figure 5.13: Homo sapiens (w)

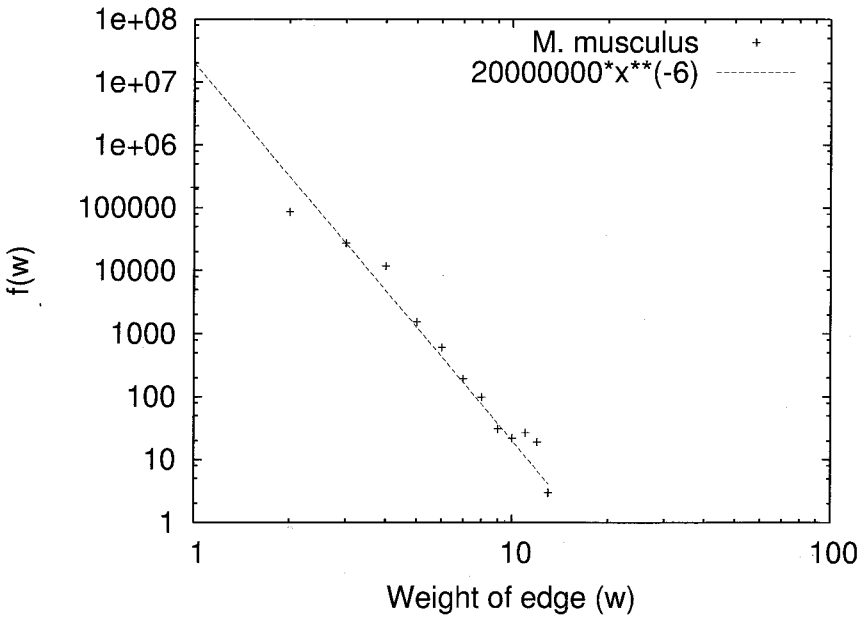

Figure 5.14: Mus musculus (w)

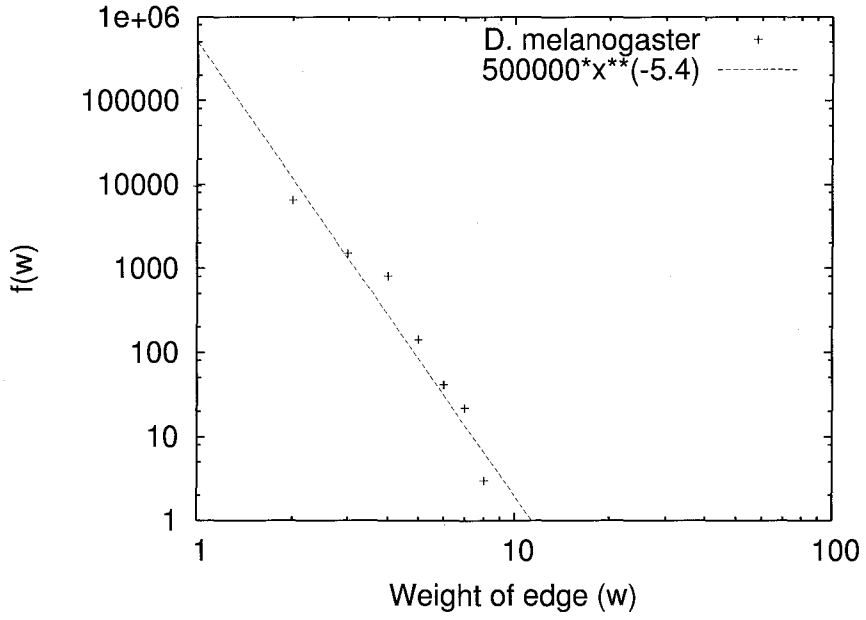

Figure 5.15: Drosophila melanogaster (w)

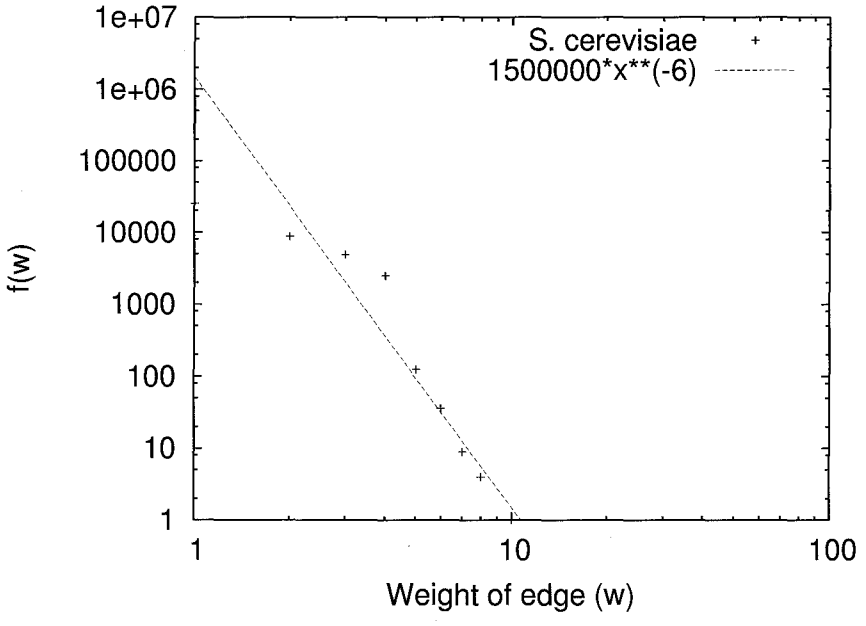

Figure 5.16: Saccharomyces cerevisiae (w)

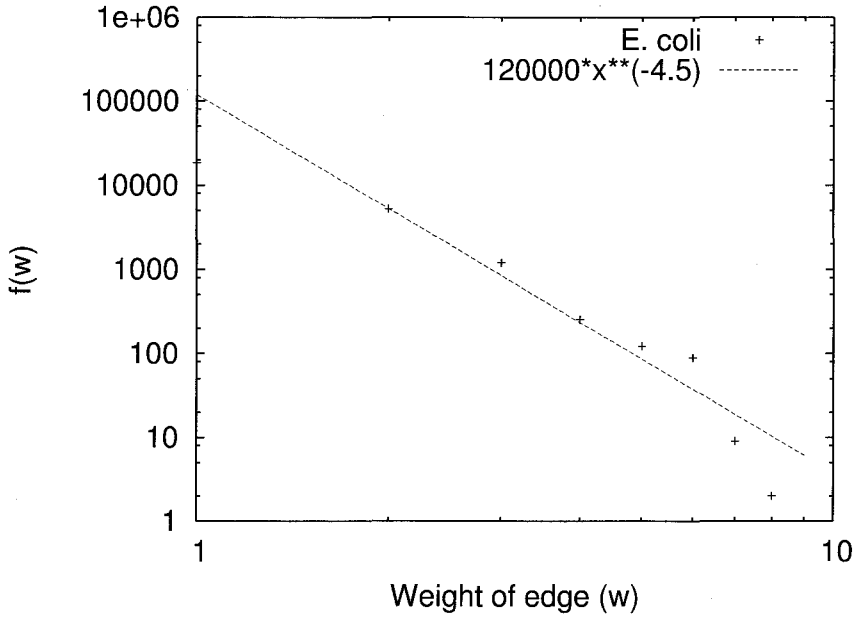

Figure 5.17: Escherichia coli (w)

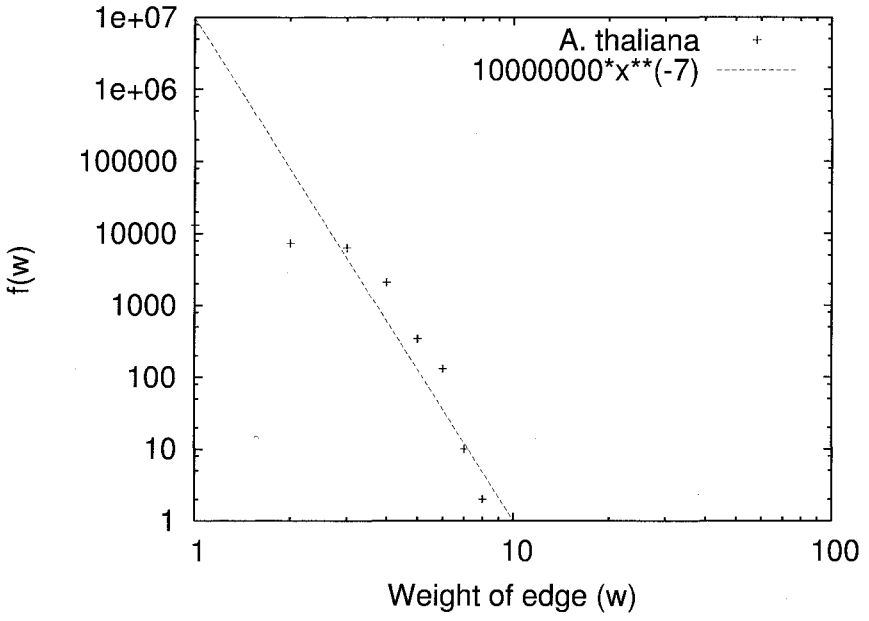

Figure 5.18: Arabidopsis thaliana (w)

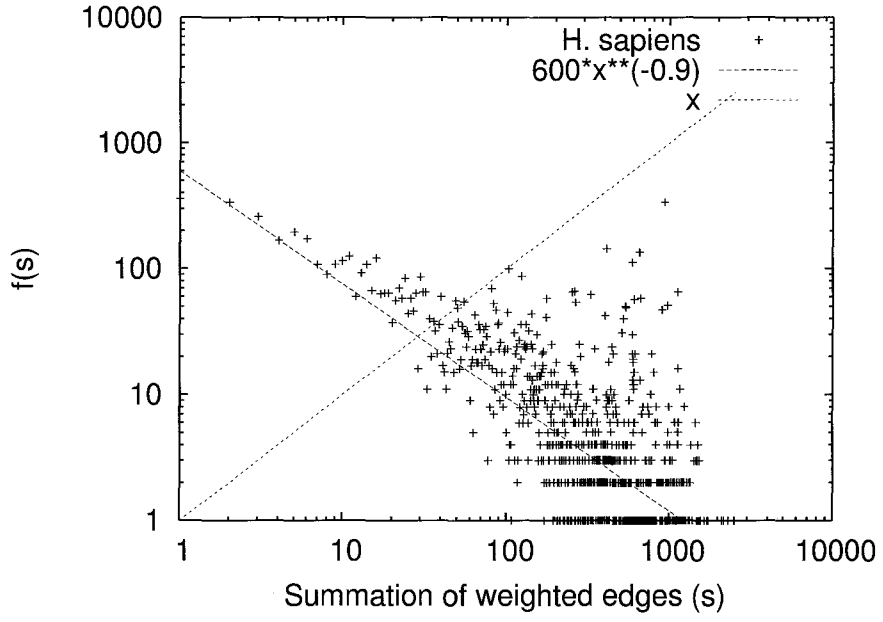

Figure 5.19: Homo sapiens (s)

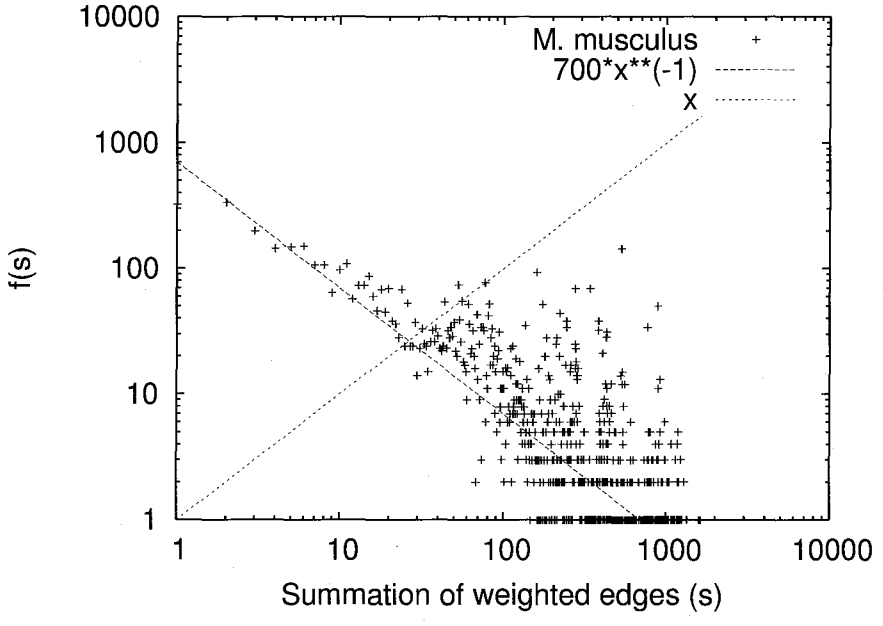

Figure 5.20: Mus musculus (s)

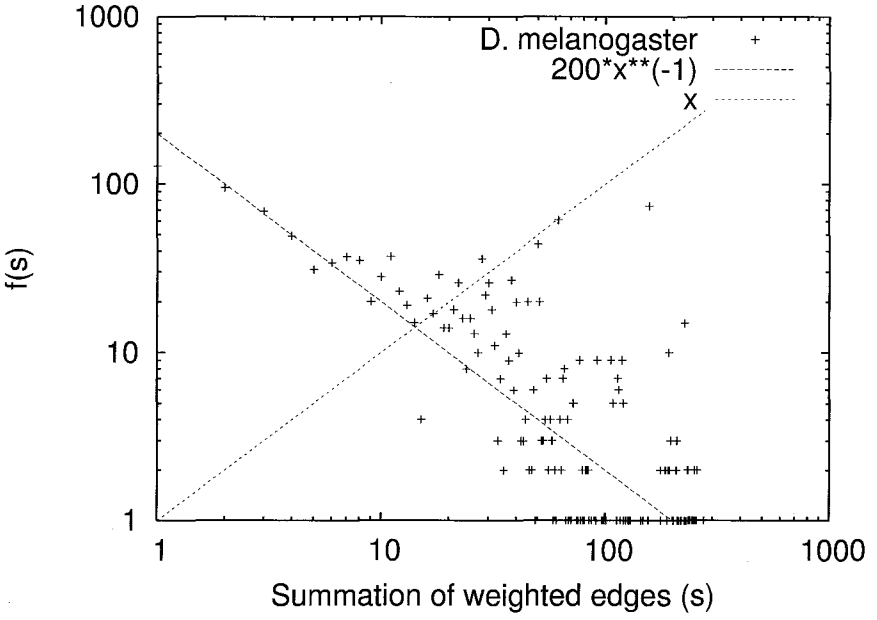

Figure 5.21: Drosophila melanogaster (s)

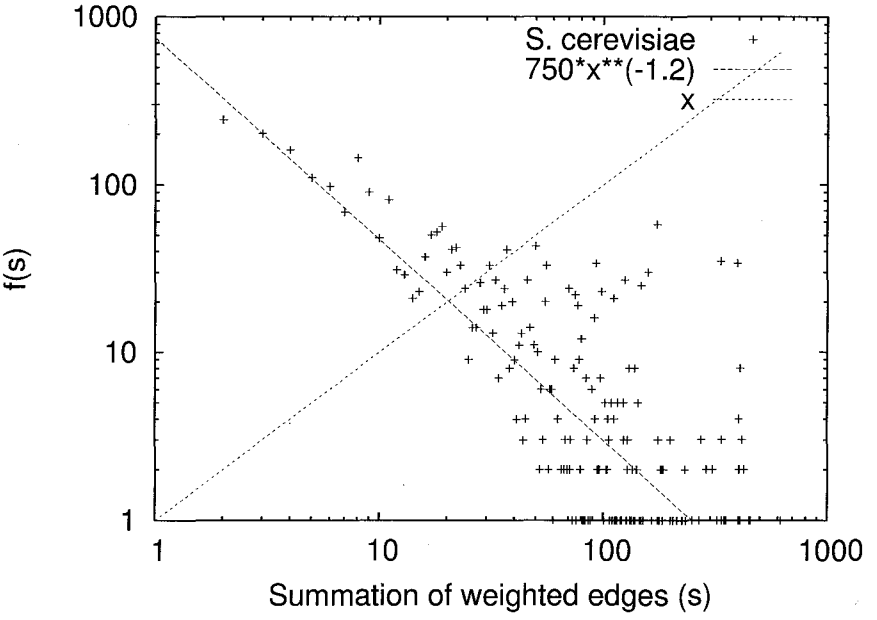

Figure 5.22: Saccharomyces cerevisiae (s)

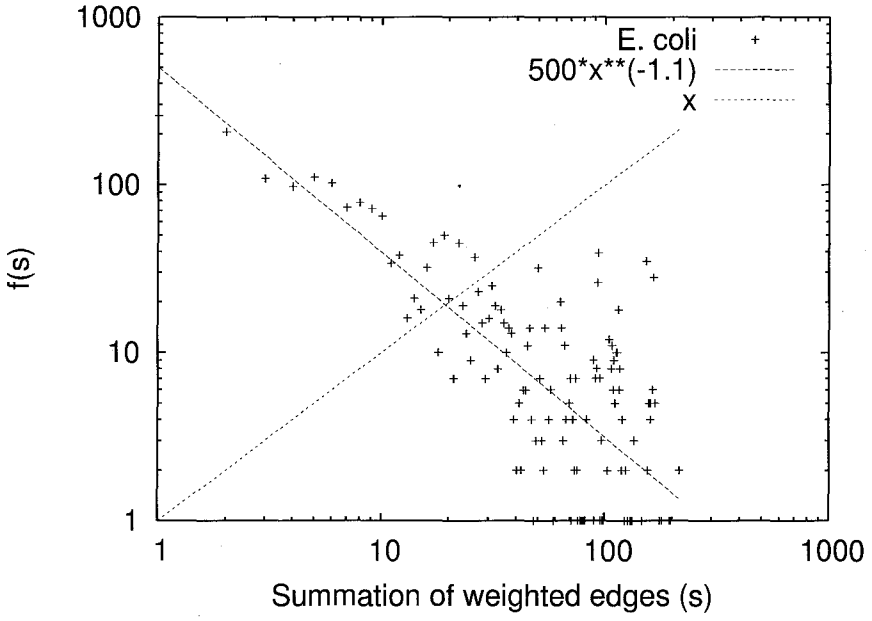

Figure 5.23: Escherichia coli (s)

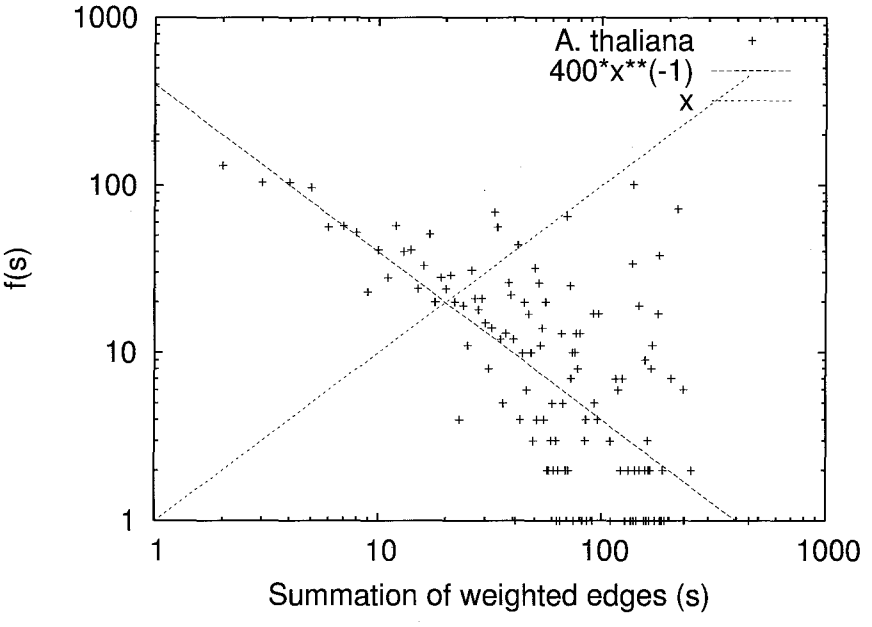

Figure 5.24: Arabidopsis thaliana (s)

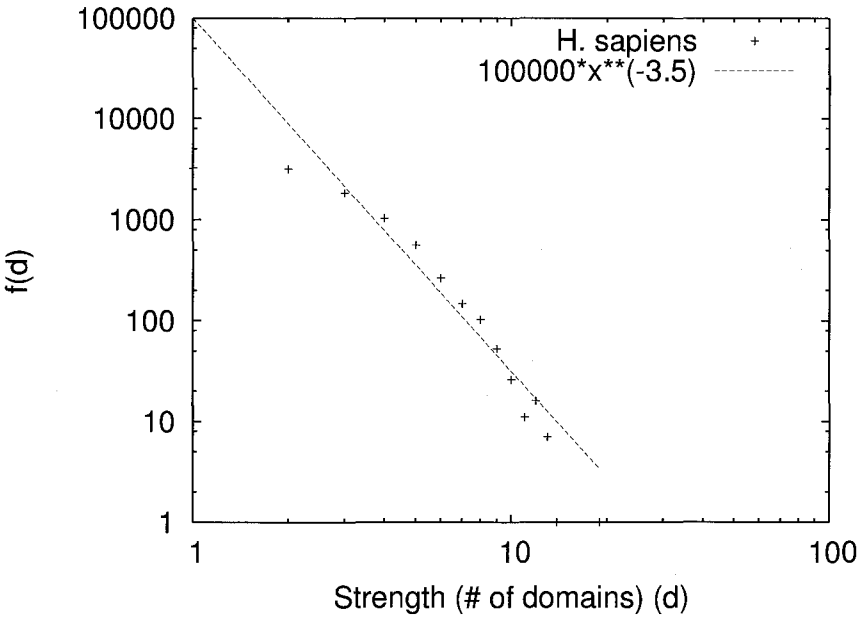

Figure 5.25: Homo sapiens (d)

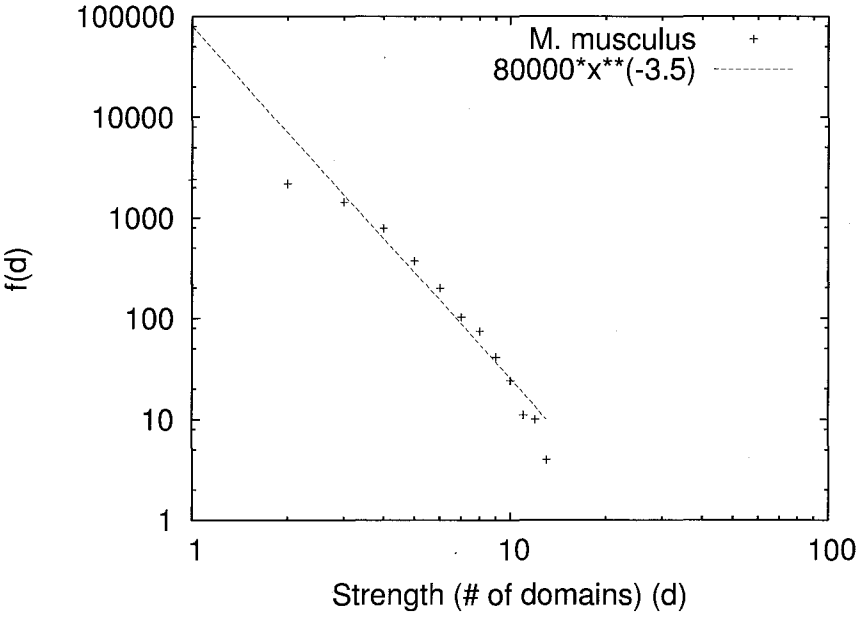

Figure 5.26: Mus musculus (d)

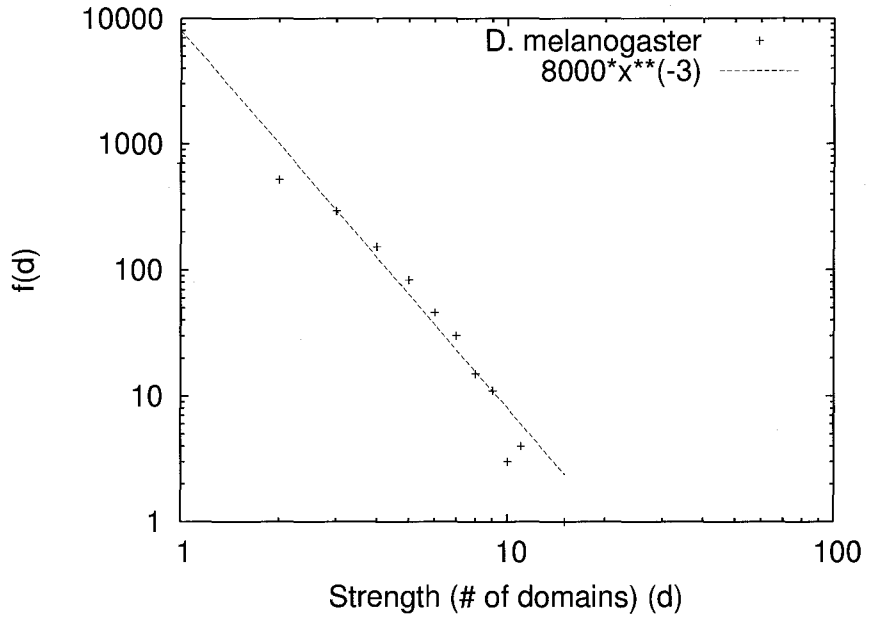

Figure 5.27: Drosophila melanogaster (d)

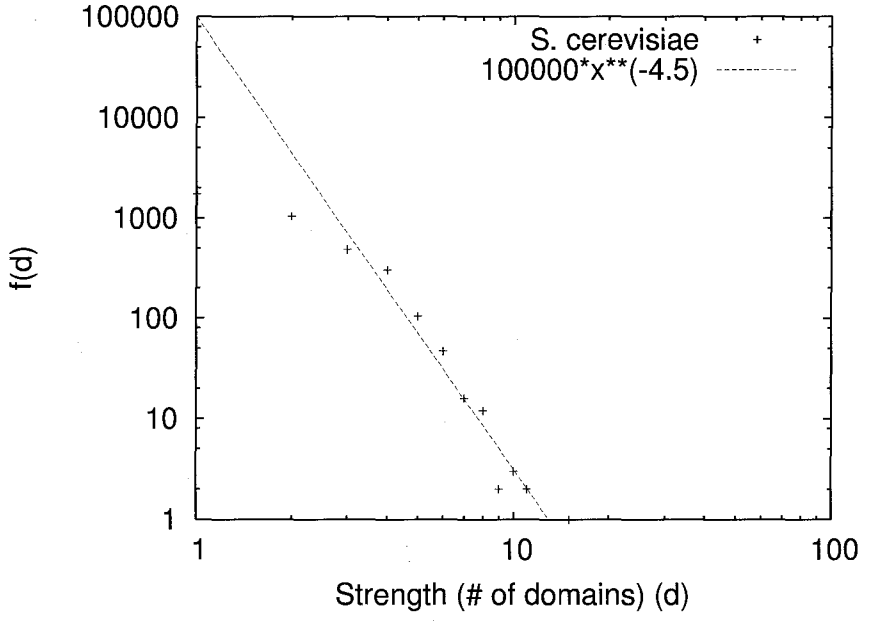

Figure 5.28: Saccharomyces cerevisiae (d)

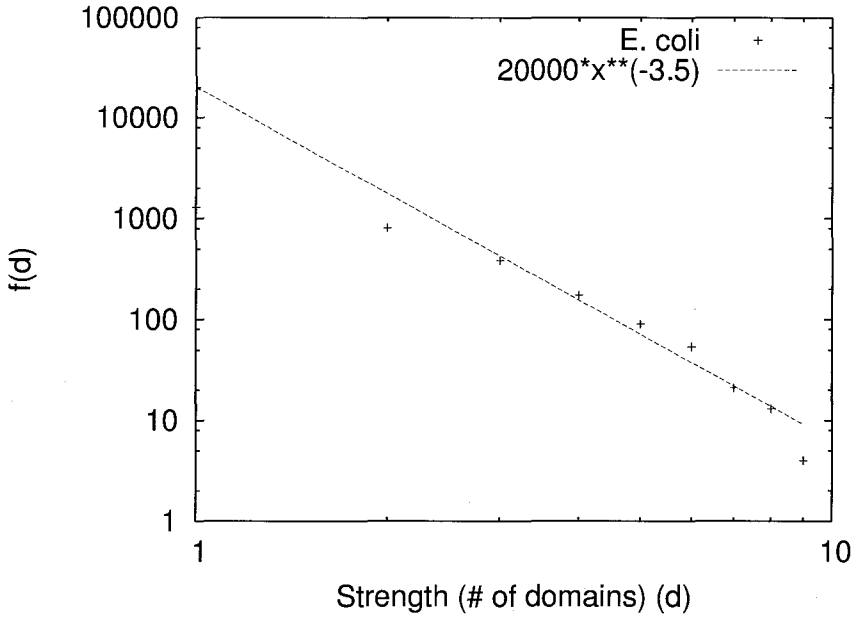

Figure 5.29: Escherichia coli (d)

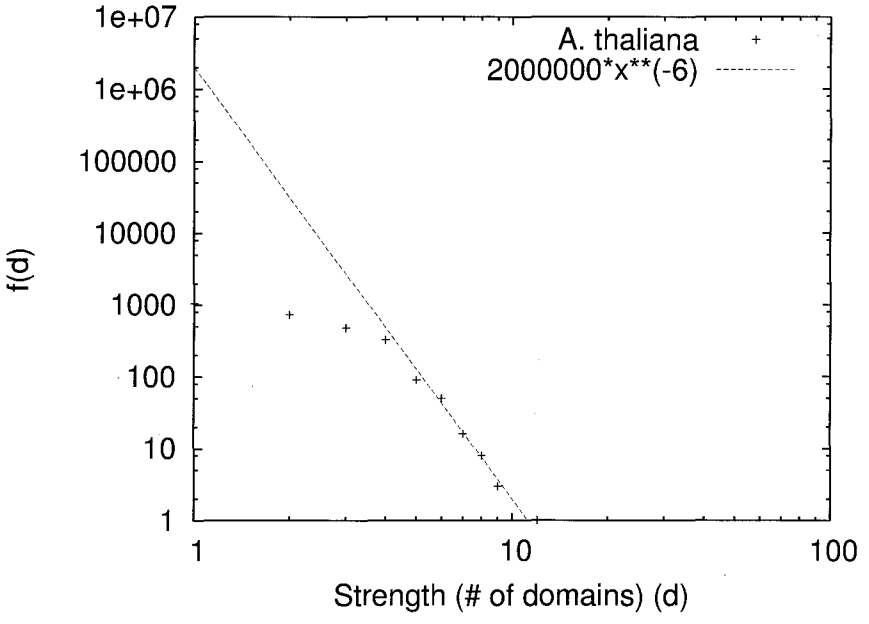

Figure 5.30: Arabidopsis thaliana (d)

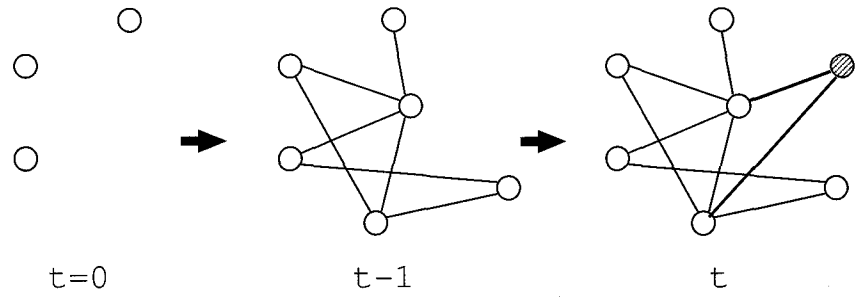

Figure 5.31: BA model. First, there are n_0 (= 3) vertices at $t = 0$. At every timestep, a new vertex and m (= 2) edges are added.

5.2 Protein Evolution Model

The BA model proposed by Barabási and Albert[5, 6] is a model which explains scale-free behaviors. We briefly review the BA model before I propose models which explain the two types of power-law behaviors of protein domain networks.

5.2.1 BA Model

The BA model is defined as a model of growing networks with the following two properties (see Figure 5.31):

(1) Growth: Starting with a small number (n_0) of vertices, at every timestep we add a new vertex with $m(\leq n_0)$ edges (that will be connected to the vertices already present in the system).

(2) Preferential attachment: When choosing the vertices to which the new vertex connects, we assume that the probability Π that a new vertex will be connected to vertex i depends on the connectivity k_i of that vertex, such that

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j}. \tag{5.2}$$

In this model, after t timesteps, a random network has $(n_0 + t)$ vertices and mt edges. We will confirm that this network is a scale-invariant state,

that is, $P(k)$ follows a power law with an exponent, and the scaling exponent is independent of m. The vertices that have the most connections are those that have been added at the early stages of the network development because these vertices can connect to more vertices than the vertices added later. Thus, some of the oldest vertices have a very long time to acquire links, and it appears at the high-k part of $P(k)$. The time dependence of the connectivity of a given vertex can be calculated analytically using a mean-field approach. Barabási and Albert assume that k is continuous, and thus the probability $\Pi(k_i) = \frac{k_i}{\sum_j k_j}$ can be interpreted as a continuous rate of change of k_i. Consequently, we can write for a vertex i on the timestep t

$$\begin{aligned} \frac{\partial k_i}{\partial t} &= E(\#\text{ of edges which vertex } i \text{ will obtain on the timestep } t) \\ &= m\Pi(k_i) = m\frac{k_i}{\sum_j k_j}. \end{aligned} \tag{5.3}$$

Because the summation of degrees is equal to double the number of added edges, and we consider t and k_js as continuous values, we obtain $\sum_j k_j = 2mt$. Taking into account this equation, we have

$$\frac{\partial k_i}{\partial t} = \frac{k_i}{2t}. \tag{5.4}$$

The solution of this equation, with the initial condition that vertex i was added to the system at time t_i $(= i - n_0)$ with connectivity $k_i(t_i) = m$, is

$$k_i(t) = m\sqrt{\frac{t}{t_i}}. \tag{5.5}$$

The probability that a vertex i has a connectivity $k_i(t)$ smaller than k can be written as

$$P(k_i(t) < k) = P\left(t_i > \frac{m^2 t}{k^2}\right) \tag{5.6}$$

$$= P\left(\frac{m^2 t}{k^2} < t_i \leq t\right). \tag{5.7}$$

Since $n_0 + t$ vertices are added uniformly at random, we have

$$P\left(\frac{m^2 t}{k^2} < t_i \leq t\right) = \int_{\frac{m^2 t}{k^2}}^{t} P(t_i) dt_i \tag{5.8}$$

$$= \frac{1}{n_0+t}\left(t - \frac{m^2 t}{k^2}\right) \tag{5.9}$$

$$= \frac{t}{n_0+t} - \frac{m^2 t}{k^2(n_0+t)}. \tag{5.10}$$

Therefore, the probability density for $P(k)$ is

$$P(k) = \frac{\partial P(k_i(t) < k)}{\partial k} = \frac{2m^2 t}{k^3(n_0+t)} \propto k^{-3}. \tag{5.11}$$

It follows the power law of $\gamma = 3$.

5.2.2 Model of One Domain within One Protein

First, I propose a simple model of protein domain networks. We call this model the one-domain model. In the next section, I will propose an extended model of the one-domain model. In this model, we consider exactly only one domain within one protein. The growing procedure of this model is as follows,

(1) Start with no protein and no domain. We suppose the time $t = 1$ when the first domain is created.

(2) Mutation: create a new protein that consists of a new domain with probability $(1 - a)$

(3) Duplication: create a new protein that consists of an existing domain with probability a. The duplicated original domain is uniformly at random selected from all of the existing proteins.

Let $n^{(i)}$ be the number of i-th domain. After t timesteps, a random network following this model has t proteins. Considering the probability that a protein with domain i is created, we have

$$\frac{\partial n^{(i)}}{\partial t} = a\frac{n^{(i)}}{\sum_i n^{(i)}} = a\frac{n^{(i)}}{t}. \tag{5.12}$$

The solution of this equation, with the initial condition that first domain i was created in the system at time t_i, is

$$n^{(i)} = \left(\frac{t}{t_i}\right)^a. \tag{5.13}$$

In the same way as the BA model, the probability that all the i-th domains are included in less than n proteins is

$$P(n^{(i)} < n) = P\left(t_i > \frac{t}{n^{\frac{1}{a}}}\right) \tag{5.14}$$

$$= P\left(\frac{t}{n^{\frac{1}{a}}} < t_i \leq t\right) \tag{5.15}$$

$$= \frac{1}{t}\left(t - \frac{t}{n^{\frac{1}{a}}}\right) \tag{5.16}$$

$$= 1 - \frac{1}{n^{\frac{1}{a}}}. \tag{5.17}$$

Therefore, the probability distribution $P(n)$ of the frequency of domains having n copies is

$$P(n) = \frac{\partial P(n^{(i)} < n)}{\partial n} \tag{5.18}$$

$$= \frac{1}{a} n^{-1-\frac{1}{a}}. \tag{5.19}$$

This equation means that $P(n)$ shows a negative power law.

In addition, we can derive from this distribution that the protein domain networks for some species show a positive power law. The reason is as follows. For sufficiently large n, the number of domains having n copies is expected to be

$$\frac{t}{an^{1+\frac{1}{a}}} \leq 1. \tag{5.20}$$

For such domains, the number of proteins with the identified domain is $k+1$ if the degree of the vertex which corresponds to the protein is k. Therefore, we have that the distribution follows the power law with a positive exponent close to one for these proteins.

Next, I explain another feature of a power law with a negative exponent. From Equation (5.19), taking into account that the degree k_i of $n^{(i)}$ proteins with domain i is $n^{(i)} - 1$, the probability distribution $P(k)$ is as follows,

$$P(k) = nP(n-1) \tag{5.21}$$

$$= (k+1)\frac{1}{a}k^{-1-\frac{1}{a}} \simeq \frac{1}{a}k^{-\frac{1}{a}}. \tag{5.22}$$

This model has some features and advantages. It is reasonable that it is based on well-known fundamental mechanisms such as protein mutation and duplication from an evolutional point of view of proteins. It has two types of power laws with positive and negative exponents. The exponent of the negative power law in Equation (5.22) can be modified by tuning the rate a of duplication to mutation. It generates the scale-free distribution such as the BA model without requiring any preferential attachment required in the BA model explicitly.

5.2.3 Extended Model

Actually, not only one domain, but also several domains are contained in a protein. I add a mechanism to increase the number of domains from the previous one-domain model as follows (see Figure 5.32):

(1) Start with no protein and no domain. We suppose the time $t = 1$ when the first domain is created.

(2) Mutation: create a new protein that consists of a new domain with probability $(1 - a - b)$

(3) Duplication: create a new protein that consists of all domains of an existing protein, with probability a. The duplicated original protein is uniformly at random selected from all of the existing proteins.

(4) Fusion: create a new protein that consists of all domains of an existing protein and a domain of another existing protein, with probability b. The duplicated original proteins are uniformly at random selected from all of the existing proteins.

Note that this extended model is equivalent to the one-domain model when the number of domains is limited to one, or the probability of fusion is $b = 0$.

5.2.4 Computational Experiment

I performed some computational experiments using our models. First, I set the probability parameter $b = 0$ in order to observe behaviors of the one-

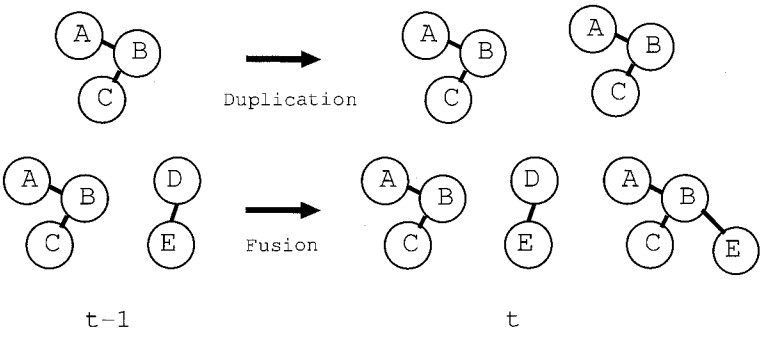

Figure 5.32: Procedures of the extended model; Duplication and fusion. Alphabetical characters represent domains. In duplication, all domains in a protein are duplicated. In fusion, one fused domain is selected from an existing protein at random, and the domain is added to another existing protein.

domain model, and performed $t = 10000$ timesteps. Figure 5.33 shows their results.

In all the cases of $a = 0.2, 0.5, 0.8, 0.95$, we see that the frequency distributions show two types of power-law tendencies. They have the tendency that the slopes of negative power are steeper for smaller a, which is the rate of duplication to mutation. In other words, more proteins are created by mutations. For positive power laws, the distributions of larger a obtain proteins with larger degrees.

Figure 5.34 shows the result of a simulation in the extended model. This result also shows that the extended model has two types of power-law tendencies. We see that the shape is quite similar to the results of real data.

5.3 Discussion

It is well known that many biological and other networks show power-law behaviors. The probability distributions of degree k have almost always been a negative exponent. Those of our protein domain networks show power laws with a positive exponent.

Table 5.2, 5.3, 5.4, 5.5, 5.6 and 5.7 show main domains within proteins along almost positive power laws for some species: H. sapiens, M. muscu-

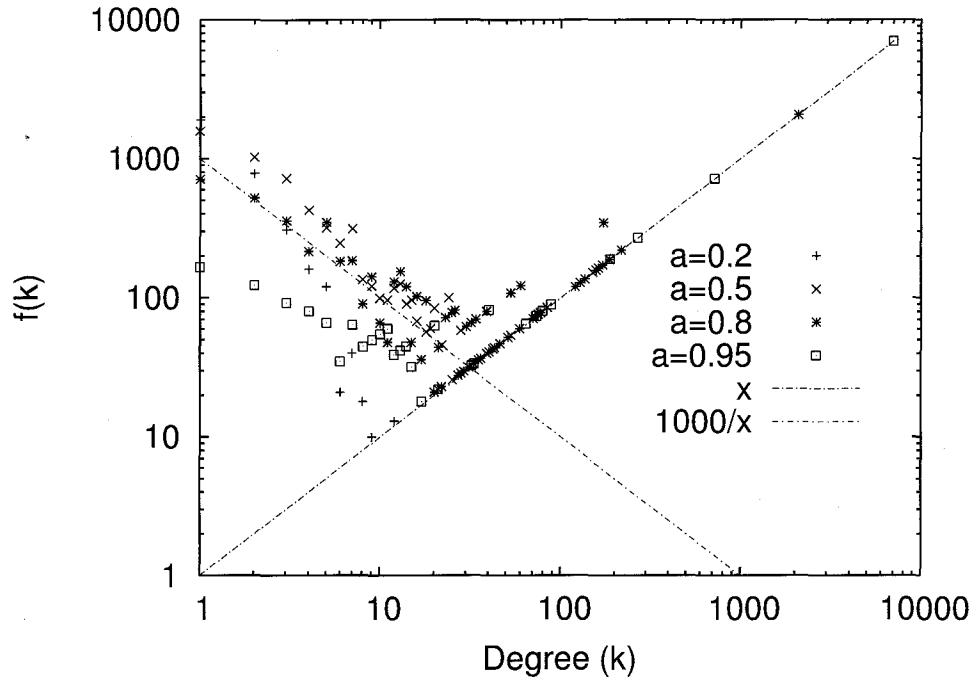

Figure 5.33: Results of simulation of one-domain model ($b = 0$) when $a = 0.2, 0.5, 0.8, 0.95$, respectively.

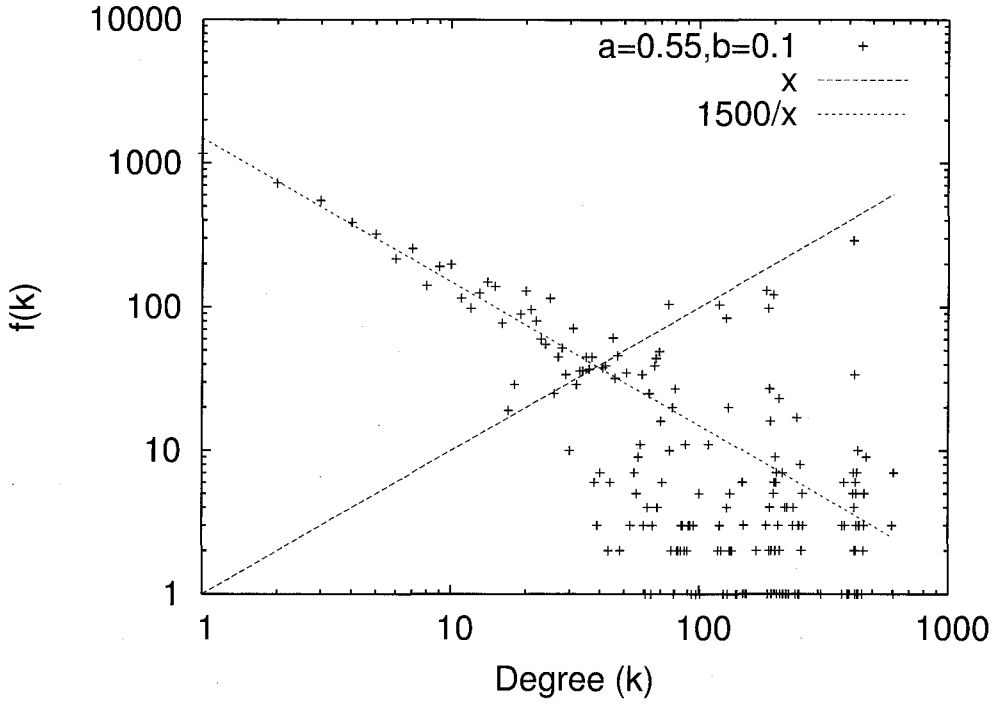

Figure 5.34: Result of a simulation in the extended model. $a = 0.55, b = 0.1$.

Table 5.2: Main Pfam domains of Homo sapiens within proteins along almost positive power laws.

k	$f(k)$	Domain (frequency)	Function
588	582	PF00001(582)	7 transmembrane receptor (rhodopsin family)
464	300	PF00047(299)	Immunoglobulin domain
398	208	PF00096(206)	Zinc finger (C2H2 type)
352	195	PF00069(195)	Protein kinase domain
174	155	PF00046(153)	Homeobox domain

lus, D. melanogaster, S. cerevisiae, E. coli, and A. thaliana. The domains are 7 transmembrane receptor, immunoglobulin domain, protein kinase domain, and so on. Their common features are that they have variety in their functions and sequences. For example, one of the G-protein-coupled receptors, 7 transmembrane receptors represent a widespread protein family whose functions include hormone, neurotransmitter, and light receptors. All of them transduce extracellular signals through interaction with G proteins. Although their activating ligands vary widely in structure and character, the amino acid sequences of the receptors are very similar and are believed to adopt a common structural framework comprising 7 transmembrane helices[9].

Like these domains, some kinds of domains have obtained a novel function without changing their entire main structure. As a result, it enables cells to receive various extracellular signals because there are various receptors that correspond to their signals.

Table 5.3: Main Pfam domains of Mus musculus within proteins along almost positive power laws.

k	$f(k)$	Domain (frequency)	Function
343	252	PF00047(252)	Immunoglobulin domain
277	155	PF00069(155)	Protein kinase domain
255	251	PF00001(251)	7 transmembrane receptor (rhodopsin family)
172	151	PF00046(151)	Homeobox domain
146	101	PF00096(101)	Zinc finger (C2H2 type)

Table 5.4: Main Pfam domains of Drosophila melanogaster within proteins along almost positive power laws.

k	$f(k)$	Domain (frequency)	Function
83	85	PF00067(84)	Cytochrome P450
70	45	PF00069(45)	Protein kinase domain
62	63	PF02949(63)	7tm Odorant receptor
51	45	PF00046(45)	Homeobox domain
45	32	PF00096(32)	Zinc finger (C2H2 type)

Table 5.5: Main Pfam domains of Saccharomyces cerevisiae within proteins along almost positive power laws.

k	$f(k)$	Domain (frequency)	Function
104	93	PF00069(93)	Protein kinase domain
74	70	PF00400(70)	WD domain (G-beta repeat)
41	45	PF00083(42)	Sugar transporter
37	34	PF00096(32)	Zinc finger (C2H2 type)
33	31	PF00004(31)	ATPase family (AAA)

Table 5.6: Main Pfam domains of Escherichia coli within proteins along almost positive power laws.

k	$f(k)$	Domain (frequency)	Function
61	59	PF00005(59)	ABC transporter
40	41	PF00419(41)	Fimbrial protein
31	30	PF00126(30)	LysR family
28	29	PF00083(29)	Sugar transporter
26	30	PF00165(27)	AraC family

Table 5.7: Main Pfam domains of Arabidopsis thaliana within proteins along almost positive power laws.

k	$f(k)$	Domain (frequency)	Function
73	74	PF00141(74)	Peroxidase
72	73	PF00067(73)	Cytochrome P450
70	68	PF03106(68)	WRKY DNA-binding domain
54	42	PF00069(42)	Protein kinase domain
42	43	PF03195(43)	Protein of unknown function DUF260

Chapter 6

Conclusion and Future work

6.1 Summary

I developed some new inferring methods, a method based on linear programming for inferring protein-protein interactions (LPBN) in chapter 2, a method applied to strengths of protein-protein interactions (LPNM), and a faster method for strengths (ASNM) in chapter 4. They outperformed existing methods with respect to classification accuracy or errors.

On deriving algorithms such as the above methods, it is essential to understand how difficult the problem is from a computational point of view. In chapter 3, I defined a problem (MAX PPI) to maximize correctly classified examples, and proved that the problem is MAX SNP-hard. It means that there is no polynomial-time algorithm to approximate the problem by an arbitrary ratio. Therefore, heuristic algorithms such as the LPBN method are required.

In chapter 5, I defined a protein domain network from the point of view of protein evolution. In real data from the UniProt knowledgebase, the probability distribution of degrees in the protein domain network showed two types of power laws. I proposed models which reconstruct the distribution, showed their behaviors for a one-domain model theoretically, and performed computational experiments to verify them.

6.2 Future Directions

In order to improve the inference methods, we can consider adding other information such as subcellular localization. Localization information is useful for inferring protein-protein interactions because two proteins must localize to the same subcellular site in order to interact with each other.

The probabilistic model of protein-protein interactions studied in this thesis is too simple. Therefore, the model should be improved so that we can understand protein-protein interactions more accurately. For example, we can consider the conditional probabilities of interactions between domains, although the events of domain-domain interactions are independent from each other in the original probabilistic model.

In this thesis, I have studied only protein-protein interactions. However, we may apply the proposed LP-based methods to other kinds of interactions, for example, interactions between proteins and DNAs or RNAs.

Though I have shown the hardness of MAX PPI for binary interaction data, it is also important to understand how difficult it is to minimize errors of strengths of protein-protein interactions for numerical interaction data. If the strengths are represented by multiple values, the minimization problem is more difficult than MAX PPI because MAX PPI can be considered as the problem restricted to binary data. However, it is unclear how difficult the problem is for real numbers.

In protein domain networks, we can not deal with the event of domain shuffling. Therefore, network models in which the effects of domains are taken into account should be studied.

Although it is known that some protein-protein interaction networks are scale-free, the networks were not analyzed using protein domain compositions. Therefore, it would be interesting to analyze protein-protein interaction networks using domain information.

Bibliography

[1] Amaldi, E. and Kann, V., On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems, *Theoretical Computer Science*, 209:237–260, 1998.

[2] Apweiler, R., Bairoch, A., Wu, C.H., Barker, W.C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., Martin, M.J., Natale, D.A., O'Donovan, C., Redaschi, N. and Yeh, L.L., UniProt: the Universal Protein knowledgebase, *Nucleic Acids Research*, 32:D115–D119, 2004.

[3] Attwood, T.K., Bradley, P., Flower, D.R., Gaulton, A., Maudling, N., Mitchell, A.L., Moulton, G., Nordle, A., Paine, K., Taylor, P., Uddin, A. and Zygouri, C., PRINTS and its automatic supplement, prePRINTS, *Nucleic Acids Research*, 31(1):400–402, 2003.

[4] Bairoch, A. and Apweiler, R., The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000, *Nucl. Acids. Res.*, 28:45–48, 2000.

[5] Barabási, A.L., and Albert, R., Emergence of scaling in random networks, *Science*, 286:509–512, 1999.

[6] Barabási, A.L., Albert, R. and Jeong, H., Mean-field theory for scale-free random networks, *Physica A*, 272:173–187, 1999.

[7] Bateman, A., Birney, E., Cerruti, L., Durbin, R., Etwiller, L., Eddy, R.S., Griffiths-Jones, S., Howe, L.K., Marshall, M. and Sonnhammer, L.L.E., The Pfam protein families database, *Nucleic Acids Research*, 30:276–280, 2002.

[8] Bennet, P.K. and Mangasarian, L.O., Robust linear programming discrimination of two linearly inseparable sets, *Optimization Methods and Software*, 1:23–34, 1992.

[9] Birnbaumer, L., G proteins in signal trunsduction, *Annual Review of Pharmacology and Toxicology*, 30:675–705, 1990.

[10] Bock, R.J. and Gough, A.D., Predicting protein-protein interactions from primary structure, *Bioinformatics*, 17:455–460, 2001.

[11] Corpet, F., Servant, F., Gouzy, J. and Kahn, D., ProDom and ProDom-CG: tools for protein domain analysis and whole genome comparisons, *Nucleic Acids Research*, 28(1):267–269, 2000.

[12] Cortes, C. and Vapnik, V., Support-vector networks, *Machine Learning*, 20:273–297, 1995.

[13] Dempster, A.P., Laird, N.M. and Rubin, D.B., Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society*, B 39:1–38, 1977.

[14] Deng, M., Mehta, S., Sun, F. and Chen, T., Inferring domain-domain interactions from protein-protein interactions, *Genome Research*, 12:1540–1548, 2002.

[15] Deng, M., Chen, T. and Sun, F., An integrated probabilistic model for functional prediction of proteins, *Proc. 7th Annual International Conf. Computational Biology*, 95–103, 2003.

[16] Enright, J.A., Iliopoulos, I., Kyrpides, C.N. and Ouzounis, A.C., Protein interaction maps for complete genomes based on gene fusion events, *Nature*, 402:86–90, 1999.

[17] Gomez, M.G., Lo, H.S. and Rzhetsky, A., Probabilistic prediction of unknown metabolic and signal-transduction networks, *Genetics*, 159:1291–1298, 2001.

[18] Hayashida, M., Ueda, N. and Akutsu, T., Inferring strengths of protein-protein interactions from experimental data using linear programming, *Bioinformatics*, 19(Suppl.2):ii58–ii65, 2003.

[19] Hayashida, M., Ueda, N. and Akutsu, T., A simple method for inferring strengths of protein-protein interactions, *Genome Informatics*, 15(1):56–68, 2004.

[20] Hayashida, M., Ueda, N. and Akutsu, T., タンパク質間相互作用強度予測の高速化と困難性 (A fast method for inferring strengths of protein-protein interactions and a hardness result), *The IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* (電子情報通信学会論文誌), J88-A(1):83–90, 2005.

[21] Heath, D., Kasif, S. and Salzberg, S., Induction of oblique decision trees, *Proc. 13th International Joint Conf. Artificial Intelligence*, 1002–1007, 1993.

[22] Hulo, N., Sigrist, C.J.A., Le Saux, V., Langendijk-Genevaux, P.S., Bordoli, L., Gattiker, A., De Castro, E., Bucher, P., Bairoch, A., Recent improvements to the PROSITE database, *Nucleic Acids Research*, 32:D134-D137, 2004.

[23] Ito, T., Tashiro, K., Muta, S., Ozawa, R., Chiba, T., Nishizawa, M., Yamamoto, K., Kuhara, S. and Sakaki, Y., Towards a protein-protein interaction map of the budding yeast: a comprehensive system to examine two-hybrid interactions in all possible combinations between the yeast proteins, *Proc. Natl. Acad. Sci.*, 97:1143–1147, 2000.

[24] Ito, T., Chiba, T., Ozawa, R., Yoshida, M., Hattori, M. and Sakaki, Y., A comprehensive two-hybrid analysis to explore the yeast protein interactome, *Proc. Natl. Acad. Sci.*, 98:4569–4574, 2001.

[25] Jeong, H., Mason, S.P., Baraási, A.L. and Oltvai, Z.N., Lethality and centrality in protein networks, *Nature*, 411:41–42, 2001.

[26] Joachims, T., Making large-scale SVM learning practical, *Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.)*, MIT Press, 169–185, 1999.

[27] Kim, K.W., Park, J. and Suh, K.J., Large scale statistical prediction of protein-protein interaction by potentially interacting domain (PID) pair, *Genome Informatics*, 13:42–50, 2002.

[28] Letovsky, S. and Kasif, S., Predicting protein function from protein/protein interaction data: a probabilistic approach, *Bioinformatics*, 19(1):197–204, 2003.

[29] Letunic, I., Copley, R.R., Schmidt, S., Ciccarelli, F.D., Doerks, T., Schultz, J., Ponting, C.P. and Bork, P., SMART 4.0: towards genomic data integration, *Nucleic Acids Research*, 32(1):D142–D144, 2004.

[30] Lu, L., Arakaki, A.K., Lu, H., and Skolnick, J., Multimeric threading-based prediction of protein-protein interactions on a genomic scale: Application to the Saccharomyces cerevisiae Proteome, *Genome Research*, 13:1146–1154, 2003.

[31] Mamitsuka, H., Efficient mining from heterogeneous data sets for predicting protein-protein interactions, *Proc. 14th International Workshop Database and Expert Systems*, 32–36, 2003.

[32] Marcotte, M.E., Pellegrini, M., Ng, H., Rice, D.W., Yeates, O.T. and Eisenberg, D., Detecting protein function and protein-protein interactions from genome sequences, *Science*, 285:751–753, 1999.

[33] Marcotte, M.E., Pellegrini, M., Thompson, J.M., Yeates, O.T. and Eisenberg, D., A combined algorithm for genome-wide prediction of protein function, *Nature*, 402:83–86, 1999.

[34] Murthy, K.S., Kasif, S. and Salzberg, S., A system for induction of oblique decision trees, *J. Art. Int. Res.*, 2:1–32, 1994.

[35] Papadimitriou, C.H. and Yannakakis M., Optimization, approximation, and complexity classes, *J. Comp. Sys. Sci.*, 43:425–440, 1991.

[36] Roos, C., Terlaky, T. and Vial, J.P., *Theory and Algorithms for linear optimization*, Wiley, 1997.

[37] Sprinzak, E. and Margalit, H., Correlated sequence-signatures as markets of protein-protein interaction, *J. Mol. Biol.*, 311:681–692, 2001.

[38] Uetz, P., Giot, L., Cagney, G., Mansfield, T.A., Judson, S.R., Knight, R.J., Lockshon, D., Narayan, V., Srinivasan, M., Pochart, P., Qureshi-Emili, A., Li, Y., Godwin, B., Conover, D., Kalbfleisch, T., Vijayadamodar, G., Yang, M., Johnston, M., Fields, S. and Rothberg, M.J., A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae, *Nature*, 403:623–627, 2000.

[39] Vanderbei, J.R., Linear Programming. Foundations and Extensions, *Kluwer Academic Publishers*, Boston, 1996.

[40] Vazirani, V.V., *Approximation algorithms*, Springer-Verlag, 1998.

[41] Wagner, A. and Fell, D.A., The small world inside large metabolic networks, *Proc. R. Soc. Lond. B*, 268:1803–1810, 2001.

[42] Wojcik, J. and Schächter, C., Protein-protein interaction map inference using interacting domain profile pairs, *Bioinformatics*, 17:S296–S305, 2001.

[43] Wright, S.J., *Primal-dual interior-point methods*, SIAM, 1997.

[44] Xenarios, I., Salwínski, L., Duan, X.J., Higney, P., Kim, S. and Eisenberg, D., DIP: The database of interacting proteins. A research tool for studying cellular networks of protein interactions, *Nucl. Acids. Res.*, 30:303–305, 2002.

[45] Zdobnov, M.E. and Apweiler, R., InterProScan - an integration platform for the signature-recognition methods in InterPro, *Bioinformatics*, 17:847–848, 2001.

List of Publications by the Author

Journal Papers

1. Hayashida, M., Ueda, N. and Akutsu, T., Inferring strengths of protein-protein interactions from experimental data using linear programming, *Bioinformatics*, 19(Suppl.2):ii58–ii65, 2003.

2. Hayashida, M., Ueda, N. and Akutsu, T., タンパク質間相互作用強度予測の高速化と困難性 (A fast method for inferring strengths of protein-protein interactions and a hardness result), *The IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* (電子情報通信学会論文誌), J88-A(1):83–90, 2005.

3. Sato, I., Hayashida, M., Kai, F., Sato, Y. and Ikeuchi, K., 実光源下での画像生成:基礎画像の線形和による高速レンダリング手法 (Fast image synthesis of virtual objects in a real scene with natural shading), *The IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* (電子情報通信学会論文誌), J84-DII(8):1864–1872, 2001.

Conference Papers

1. Hayashida, M., Ueda, N. and Akutsu, T., Inferring strengths of protein-protein interactions from experimental data using linear programming,

European Conference on Computational Biology 2003, (same as 1 of Journal Papers).

2. Hayashida, M., Ueda, N. and Akutsu, T., A simple method for inferring strengths of protein-protein interactions, *International Workshop on Bioinformatics and Systems Biology 2004*, *Genome Informatics*, 15(1):56–68, 2004.

3. Akutsu, T., Hayashida, M., Tomita, E., Suzuki, J. and Horimoto, K., Protein Threading with Profiles and Constraints, *Fourth IEEE International Symposium on Bioinformatics and Bioengineering 2004*, 537–544.

4. Sato, I., Hayashida, M., Kai, F., Sato, Y. and Ikeuchi, K., 複合現実感における光学的整合性の実現：基礎画像の線形和による高速レンダリング手法, 画像の認識・理解シンポジウム (MIRU2000), 1:107-112.